



National Library
of Canada

Bibliothèque nationale
du Canada

Canadian Theses Service

Services des thèses canadiennes

Ottawa, Canada
K1A 0N4

CANADIAN THESES

NOTICE

The quality of this microfiche is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Previously copyrighted materials (journal articles, published tests, etc.) are not filmed.

Reproduction in full or in part of this film is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30. Please read the authorization forms which accompany this thesis.

**THIS DISSERTATION
HAS BEEN MICROFILMED
EXACTLY AS RECEIVED**

THÈSES CANADIENNES

AVIS

La qualité de cette microfiche dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

Les documents qui font déjà l'objet d'un droit d'auteur (articles de revue, examens publiés, etc.) ne sont pas microfilmés.

La reproduction, même partielle, de ce microfilm est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30. Veuillez prendre connaissance des formules d'autorisation qui accompagnent cette thèse.

**LA THÈSE A ÉTÉ
MICROFILMÉE TELLE QUE
NOUS L'AVONS REÇUE**



National Library
of Canada

Bibliothèque nationale
du Canada

Ottawa, Canada
K1A 0N4

TC -

0-315-23321-4

CANADIAN THESES ON MICROFICHE SERVICE - SERVICE DES THÈSES CANADIENNES SUR MICROFICHE

PERMISSION TO MICROFILM - AUTORISATION DE MICROFILMER

• Please print or type - Écrire en lettres moulées ou dactylographier

AUTHOR - AUTEUR

Full Name of Author - Nom complet de l'auteur

KOK WAI CHAN

Date of Birth - Date de naissance

SEPT 13, 1959

Canadian Citizen - Citoyen canadien

☐ Yes / Oui

☒ No / Non

Country of Birth - Lieu de naissance

MALAYSIA

Permanent Address - Résidence fixe

40, KEGAT WAH KEONG,
WAH KEONG PARK
IPOH, PERAK
MALAYSIA

THESIS - THÈSE

Title of Thesis - Titre de la thèse

IMAGE EXPANSION USING INTERPOLATION AND
NOISE CLEANING METHODS

Degree for which thesis was presented
Grade pour lequel cette thèse fut présentée

M.Sc.

Year this degree conferred
Année d'obtention de ce grade

1985

University - Université

UNIV. OF ALBERTA

Name of Supervisor - Nom du directeur de thèse

WAYNE A. DAVIS

AUTHORIZATION - AUTORISATION

Permission is hereby granted to the NATIONAL LIBRARY OF CANADA to
microfilm this thesis and to lend or sell copies of the film.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.

L'autorisation est, par la présente, accordée à la BIBLIOTHÈQUE NATIONALE DU CANADA de microfilmer cette thèse et de prêter ou de vendre des exemplaires du film.

L'auteur se réserve les autres droits de publication; ni la thèse ni de longs extraits de celle-ci ne doivent être imprimés ou autrement reproduits sans l'autorisation écrite de l'auteur.

ATTACH FORM TO THESIS - VEUILLEZ JOINDRE CE FORMULAIRE À LA THÈSE

Signature

Chan Kok Wai

Date

OCT 11, 1985

The University of Alberta

Image Expansion using Interpolation and Noise-Cleaning Methods

by



Kok Wai Chan

A thesis
submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree
of Master of Science

Department of Computing Science

Edmonton, Alberta
Fall, 1985

THE UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR: Kok Wai Chan

TITLE OF THESIS: Image Expansion using Interpolation and Noise-Cleaning
Methods

DEGREE FOR WHICH THIS THESIS WAS PRESENTED: Master of Science

YEAR THIS DEGREE GRANTED: 1985

Permission is hereby granted to The University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.

(Signed) Chan Wai Chan

Permanent Address:

40, Regat Wah Keong
Wah Keong Park
Ipoh, Perak
Malaysia

Dated October 10, 1985.

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research, for acceptance, a thesis entitled **Image Expansion using Interpolation and Noise-Cleaning Methods** submitted by **Kok Wai Chan** in partial fulfillment of the requirements for the degree of **Master of Science**.

Wayne A. Daws

Supervisor

S. Cabay

Z. J. Koles

Thelma

W. J. G. G. G.

Date October 7, 1985.

ABSTRACT

This thesis is concerned with the problem of providing reasonable renditions of low resolution data using high resolution display devices. The applicability of interpolation and heuristics exemplified by noise cleaning methods are investigated to achieve such renditions. In this research, bilinear and certain piecewise bicubic interpolation are studied in both the spatial and frequency domains. Although in theory the higher degree interpolations provide smoother results, the discrete nature of images hinders this in applications.

The use of noise-cleaning techniques is also proposed for the expansion of images. Due to the lack of mathematical tools in this area, no universally accepted quantitative measure exists for image evaluation. Some simple measures are studied to evaluate the expansion techniques based on the experimental results.

Acknowledgements

I am grateful to Dr. Davis for his encouragement, understanding and guidance without which this thesis could never have come to fruition. It is also my pleasure to thank Drs. Cabay, Caelli, Koles and Szafron for their critical reviews and helpful suggestions.

I also wish to acknowledge the assistance from Martin Dubetz who showed me how to use the IIS, Prahat Sahay who helped me in preparing this thesis, and Joanne Hodgson who typed the first draft.

Last but not least, I am grateful to Boon and my family for their emotional support.

Table of Contents

Chapter	Page
Chapter 1: INTRODUCTION	1
1.1. Fundamentals, Definitions and Assumptions	1
1.2. Aliasing	5
1.3. Problem Definition and Approaches	6
1.4. Thesis Organization	11
Chapter 2: Expansions	12
2.1. Scaling	12
2.2. Interpolation in the Frequency Domain	14
2.2.1. Preliminary Definitions and Notations	14
2.2.2. Definitions, Properties and Useful Results of the Fourier Transform	16
2.2.3. Sampling	19
2.3. Interpolation in the Spatial Domain	23
2.3.1. Bilinear Interpolation	24
2.3.2. Piecewise Cubic Interpolation	27
2.3.3. Piecewise Bicubic Interpolation	33
2.4. Fourier Transforms of the Basis Functions of the Interpolation	42
2.4.1. Linear	42
2.4.2. Piecewise Cubic Interpolation	43
2.4.3. Two Dimensional Interpolation	45
Chapter 3: Smoothing Techniques	46
3.1. Spatial Domain Smoothing and Noise-Cleaning Methods	47
3.1.1. Weighted Averaging	47
3.1.2. Local Averaging with Thresholding	49
3.1.3. Variable Neighborhood Averaging	50
3.1.4. Selective Neighbor Averaging	51
3.1.5. Maximum Homogeneity Smoothing	51
3.1.6. Maximum Homogeneity Smoothing with Masks	52
3.1.7. Mode and Median Filtering	53
3.1.8. Edge or Line Smoothing	54
3.1.9. Half Neighborhood Methods	55
3.1.10. Histogram Smoothing	56

3.1.11. Directional Smoothing	57
3.1.12. Gradient Smoothing	58
3.1.13. Laplacian Smoothing	58
3.1.14. Directionally Sensitive Weighted Averaging	59
3.2. Adaptive Smoothing and Noise-Cleaning Techniques	62
Chapter 4: IMAGE SHARPENING	64
4.1. Gradient	64
4.2. Laplacian	66
Chapter 5: RESULTS AND ANALYSIS	68
5.1. Pixel Replication.	69
5.2. Interpolations.	69
5.2.1. Bilinear Interpolation.	70
5.2.2. Piecewise Hermite Interpolation	71
5.2.3. Bicubic Spline Interpolation.	72
5.3. Smoothing and Noise-Cleaning Techniques.	73
5.3.1. Neighborhood Averaging.	74
5.3.2. Weighted Averaging.	75
5.3.3. Variable Neighborhood Averaging.	75
5.3.4. Selective Neighbor Averaging.	76
5.3.5. Tsuji's and Nagao's Schemes.	76
5.4. Mode and Median Filtering.	77
5.4.1. Half Neighborhood Methods.	78
5.4.2. Histogram Smoothing.	78
5.4.3. Directional Smoothing.	79
5.4.4. Gradient Smoothing.	79
5.5. Sharpening.	79
5.6. Performance Measures	80
Chapter 6: CONCLUSION	107
6.1. Overview	107
6.2. Summary	107
6.3. Application	110
6.4. Further Research	111
References	113

List of Tables

Table	Page
2.1 Fourier Transforms of Some Functions.	17
2.2 The Required Initial Conditions	39
5.1 Implemented Methods	89

List of Figures

Figure	Page
1.1 N8(P), the eight neighbors of P.	2
1.2 The proximity of A and B is not reflected in the chain code.	4
1.3 Gap filling algorithms based on weighted averaging.	7
2.1 Terminology of a Linear System.	19
2.2 Sampling: Spatial and Freq. Domains.	20
2.3 Bilinear Interpolation	25
2.4 Replication of Pixels	25
2.5 (Bi) Linear Interpolation	26
2.6 Guiding Points.	37
3.1 Mask with weighting defined in (3.2)	48
3.2 Mask with weighting defined in (3.3)	48
3.3 5 3×3 neighborhoods centered at (x,y)	52
3.4 Pentagonal and Hexagonal Masks	53
3.5 Median Filtering	53
3.6 The twelve possible edge/line configuration.	59
3.7 The corresponding weight matrices.	60
3.8 Weight Matrices.	62
5.1 A Quantitative Comparison of the Expansion Techniques based on (5.9).	82
5.2 A Quantitative Comparison of the Expansion Techniques based on (5.10).	83
5.3 A Quantitative Comparison of the Expansion Techniques based on (5.11).	83
5.4 A Quantitative Comparison of the Expansion Techniques based on (5.12).	84
5.5 A Quantitative Comparison of the Expansion Techniques based on (5.13).	84
5.6 A Quantitative Comparison of the Expansion Techniques based on (5.14).	85
5.7 A Quantitative Comparison of the Expansion Techniques based on (5.15).	85
5.8 A Quantitative Comparison of the Expansion Techniques based on (5.16).	86
5.9 A Quantitative Comparison of the Expansion Techniques based on	

(5.17).	86
5.10 A Quantitative Comparison of the Expansion Techniques based on	
(5.18).	87
6.1 Block Diagram: Some techniques for expansion.	110

List of Photographic Plates

Plate	Page
5.1 Replication: FIGURE, MOON and TEXT.	90
5.2 Bilinear Interpolation.	91
5.3 Modified Bilinear.	91
5.4 Piecewise Bicubic Hermite using $\Delta_1 f$	92
5.5 Piecewise Bicubic Hermite using $\Delta_2 f$	92
5.6 Piecewise Bicubic Hermite using $\Delta_3 f$	93
5.7 Complete Spline using $\Delta_1 f$	93
5.8 Complete Spline using $\Delta_2 f$	94
5.9 Complete Spline using $\Delta_3 f$	94
5.10 Natural Spline using $\Delta_3 f$	95
5.11 Averaging using a 3×3 neighborhood.	95
5.12 Averaging using a circular neighborhood of radius 5.	96
5.13 Weighted Averaging using Mask #1.	96
5.14 Weighted Averaging using Mask #2.	97
5.15 'Layered' Weighted Averaging using Mask #1.	97
5.16 'Layered' Weighted Averaging using Mask #2.	98
5.17 Variable Neighborhood Scheme #1.	98
5.18 Variable Neighborhood Scheme #2.	99
5.19 Selective Neighborhood Averaging.	99
5.20 Histogram Smoothing.	100
5.21 Tsuji et.al. Noise-Cleaning Approach.	100
5.22 Nagao et.al. Noise-Cleaning Approach.	101
5.23 'Layered' Tsuji et.al. Method.	101
5.24 'Layered' Nagao et.al. Method.	102
5.25 Mode Filtering.	102
5.26 Median Filtering.	103
5.27 Half Neighborhood Noise-Cleaning Method #1.	103
5.28 Half Neighborhood Noise-Cleaning Method #2.	104
5.29 Gradient Smoothing.	104
5.30 TEXT: Smoothing followed by sharpening.	105
5.31 FACE: Smoothing followed by sharpening.	105
5.32 MOON: Smoothing followed by sharpening.	106

—5.33 TEXT,FACE,MOON: Smoothing followed by sharpening.	106
--	-----

Chapter 1

INTRODUCTION

This thesis is concerned with the problem of providing reasonable representations for low resolution data on high resolution devices. It is natural to look for a scheme that can use currently abundant low resolution image data to obtain more visually acceptable images on high resolution devices. Also, if absolute representations are not a major concern in image display, storage efficiency can be achieved by storing the lower resolution versions of images. Transformations into higher resolution images can then be made when needed. Similarly, data transfer efficiency can be improved by transmitting lower resolution image data. Furthermore, when zooming, more pleasing displays which do not have a blocked appearance as those obtained by direct replication of pixels can also be constructed. Additionally, this capability allows the user to browse through a set of images which resemble the original pictures, but are constructed from lower resolution data.

Before discussing the details of the methods for rendering "more pleasing" images, some basic terminology and assumptions are reviewed in the next section.

1.1. Fundamentals, Definitions and Assumptions

An image refers to a two dimensional image function or array of intensity values, $f(x,y)$ where x and y denote the spatial coordinates and the value of f at any point (x,y) is proportional to the brightness (or gray level) of the image at that point. These values are real, non-negative and bounded (i.e., brightness cannot be arbitrarily large.) In addition, they are spatially bounded by a finite rectangular

region.

A digital image is an image $f(x,y)$ which has been quantized both in spatial coordinates and in brightness. The digitization involves partitioning the analog picture into compact and convex subsets of points from the analog picture. Each subset or cell is then assigned a gray level which represents the intensity of the points in that cell of the analog picture. A digital image may be considered to be a m by n matrix whose rows and columns identify cells in the image and the corresponding matrix element, $f(i,j)$ identifies the quantized gray value at the cell (i,j) . Usually, $m = n$ and is contained in the range 2^6 to 2^{12} . In this thesis, it is assumed that $m = n$. The elements of such an image array are called *picture elements* or *pixels*. It is more common, however, to find the number of quantized levels to be between 2^4 to 2^{12} . The choice of the number of quantization levels is an interesting area of research, but it lies outside the scope of this thesis. The reader may want to refer to [2, 3, 14, 30, 32] for further discussion of those issues. If the gray level of each pixel can only assume one of two values, then the image will be called a binary image.

In addition to partitioning an image using a rectangular grid, triangular and hexagonal tessellations are sometimes discussed in the literature [32].

P_3	P_2	P_1
P_4	P	P_0
P_5	P_6	P_7

Figure 1.1 $N8(P)$, the eight neighbors of P .

For a rectangular tessellation, except at the borders of the image, each pixel P of

the array has four vertical and horizontal neighbors, and four diagonal neighbors. The 8-neighbors of a pixel P are defined to be the elements of $N8(P) = \{P_0, P_1, P_2, \dots, P_7\}$ where the pixels P_i , $0 \leq i \leq 7$, are numbered as shown in Fig. 1.1. The 4-neighbors of a point is similarly defined as the members of $N4(P) = \{P_0, P_2, P_4, P_6\}$. With this notation, the neighbor P_i , $0 \leq i \leq 7$, is at $45i^\circ$ measured counterclockwise from the positive x axis. A similar notation based on the rectangular coordinates of P is also used. If P is at (x,y) , $N8\{(x,y)\} = \{P_0, P_1, P_2, \dots, P_7\}$ and $N4\{(x,y)\} = \{P_0, P_2, P_4, P_6\}$. This notation will be used in the following discussions and the appropriate definition at each usage will be governed by the context. Although the triangular or hexagonal tessellations are less frequently employed, they have two advantages over rectangular partitioning. In the latter method, the diagonal neighbors are $\sqrt{2}$ units apart, but horizontal and vertical neighbors are one unit away. For the former two tessellations, neighbors are always one unit apart. This is due to the symmetricity of each cell which allows for the boundaries of adjacent cells in the entire tessellation to be identical. In this thesis, square tessellations are assumed.

Although an image can be stored as a matrix of intensities quantized to a given set of levels, the storage requirement can be substantial. However, all the information in an image does not always need to be stored. For example, instead of storing the entire matrix of an aerial photograph with a number of water bodies in it, only the outlines of these water bodies and their positions needs to be stored. Thus, given that the contours of the entities are known, the lineal features can be encoded. A drawback to this approach, of course, is the assumption that the pixels inside the contours can be reproduced from the contour pixels. This assumption is not severe if the intensities of the interior pixels are homogeneous or if they vary in a uniform or predictable fashion. If, in addition to these assumptions, the contour

pixels have a constant gray value, another technique such as a chain code can be used to represent the lineal features of an image.

In a square-celled quantized image, each pixel P , except the end points, of a non-intersecting contour has two neighbors, each of which is an element of $N_8(P)$. Therefore, a contour can be encoded by storing the direction when moving from one pixel to the next on a contour. When the directions are specified as multiples of 45° , the resulting sequence of octal numbers is referred to as a chain code of the contour. Each of the numbers in a chain code is called a link. For further extensions to this scheme, see [13.32]. The major disadvantage to this encoding method is that geometrically close points can be far apart in the sequence. The chain code for the contour shown in Fig. 1.2 is 11077655533565322222070 with **A** as the starting point.

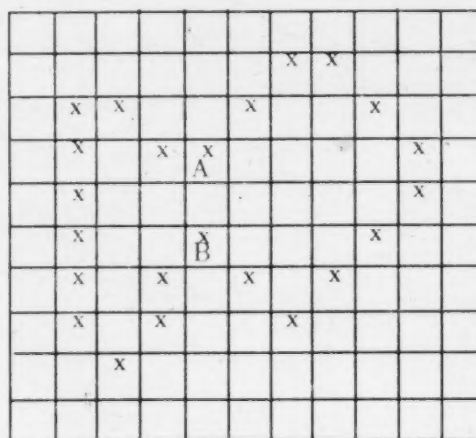


Figure 1.2 The proximity of A and B is not reflected in the chain code.

Most display screens have a rectangular tessellation. The resolution of a display screen of this type is usually defined to be the number of visible distinct line pairs per inch. An alternative definition is the number of pixels per inch on a

horizontal or vertical line. The number of rows and columns of pixels for a display, however, does not indicate the resolution of the display since the size of each pixel on the display has not been brought into consideration. Consequently, it will only make sense to describe resolution using the number of pixels per unit measure. Nevertheless, it is convenient to mean a measure for resolution when the term "number of pixels" is used and this is adopted in the rest of this thesis. With this understanding, it is obvious that the magnitudes of m and n (i.e., the number of rows and columns for an image array) determine the resolution of the digital image.

When a low resolution image is displayed on a high resolution screen instead of on a display whose screen has the same resolution as the image, the size of the image relative to the screen size is reduced. Often, the size of the displayed image is maintained by replicating each pixel in both horizontal and vertical directions the proper number of times before the image is displayed on the higher resolution screen. This direct replication of pixels is called zooming and it results in a blocked appearance of the image. Furthermore, the continuities of edges, lines and textures of regions in the image are interrupted.

In Section 1.3, this problem is examined further and other approaches to allow better renditions of low resolution images, including those that will resolve aliasing, are discussed.

1.2. Aliasing

In order to perfectly reconstruct an image, a sufficient number of discrete points must be selected from the analog picture. The sampling rate for reconstruction is governed by the sampling theorem [2, 29, 32]. When an image is undersampled, distortions occur and this phenomenon is termed aliasing. For

example, if a page of text is undersampled it will appear as a series of white and dark bands when reconstructed [29]. The same term, aliasing, is used to describe the distortions when pixel replication is performed in an image. This terminology is due mainly to the occurrence of a blocked appearance as a consequence of either replication or undersampling. In this thesis, aliasing will refer to the distortions caused by pixel replication.

Aliasing is especially prominent when an image contains lineal features. Mathematically, lines and curves have zero thickness; however, their discrete representations cannot be less than a pixel thick. Due to the discreteness of display representations and the approximations involved in drawing a line or curve, slanted edges or lines have a staircase like or jagged appearance. When a line is zoomed, each pixel is mapped into a block of pixels so that jaggies are amplified.

1.3. Problem Definition and Approaches

As mentioned earlier, when a low resolution image is displayed on a higher resolution screen, the size of the image relative to the screen size is reduced. In this thesis, the problem of displaying a more pleasing rendition of a digital enlarged image is investigated.

This problem can be solved by finding "smooth" expansions of the given data. Expansions are usually produced by replicating each pixel in the original image into a set of new pixels and this approach is termed as *zooming* when the new set of pixels takes the form of a square for equal bidirection expansions. This scheme constitutes a piecewise constant expansion and as mentioned earlier, aliasing arises from this approach. It is this effect that "smooth" expansions try to overcome when a low resolution image I is displayed in a high resolution screen without diminishing the relative size of the image actually displayed. To appreciate this, let I be an

image with resolution of r lines/inch in each of the axial directions and $D2$ be a display screen with k times the resolution of the data in both directions. If a k times bidirectional "smooth" expansion of I is available, then by spatially mapping each pixel in the expansion to a cell on $D2$, a smooth rendition of I is obtained.

The techniques for obtaining "smooth" expansions of an image can be broadly divided into four classes:

1. gap filling algorithms like interpolation.
2. pixel replication (i.e., zooming) followed by local corrections.
3. defocusing which is comprised of filtering high spectral frequencies of an image expanded by pixel replication.
4. image analysis followed by expansion.

The techniques in the first and second class can be considered as mappings of the set of pixels and their adjacent relationships of the original image I to a new set of cells on the device by initially leaving the intervening values empty and then performing a weighted average or some other scheme based on the intensities in the respective neighborhood to determine the intermediate values. This strategy is possible because each pixel is not independent of the adjacent pixels and the nature of a pixel can be estimated by the values of the pixels lying in its neighborhood.

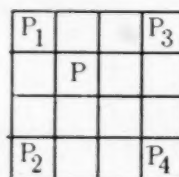


Figure 1.3 Gap filling algorithms based on weighted averaging.

In Fig. 1.3 the point P is assigned a gray level which reflects its relationship with

perhaps the pixels P_1 , P_2 , P_3 and P_4 . For pixel replication, P will have the same gray level as P_1 . It is interesting to note that the bilinear interpolation for the intensity at P is actually a weighted average with the distances between P and P_1 , P_2 , P_3 and P_4 as the weights.

Spatial smoothing techniques which are used to reduce the presence of noise in images represent the methods in the second class. Again, referring to Fig. 1.3, the gray level at P is a weighted average of the intensities of P_1 , P_2 , P_3 and P_4 , but the weights reflect the degree of confidence that P is in the same region as some of those other pixels. Typically, smoothing operations involve evaluating a number of predicates with the local information as parameters and the assigned intensity level is dependent on the outcome of these predicates. For example, the decision of whether to assign more weight to a nearby pixel value might take into consideration the variability of the regional intensities or the possibility of the occurrence of an edge.

The methods in the third class are based on the convolution theorem for Fourier Transforms. Instead of operating in the spatial domain, that is, making use of the intensity values directly in any operations, low pass filters are used in the frequency domain. The intensity values of the image is first mapped into a new domain using the Fourier transform equations and smoothing operations are performed by removing a certain range of frequencies in this new domain. The final image is then obtained by performing an inverse transformation on the processed array.

The fourth class includes methods for detecting features like edges of regions in an image and expanding these elements. A diversity of edge detection and segmentation techniques exists [24,31]. As indicated before, the detected features in an image are usually stored as a structure, i.e., its data representations

encapsulate the form of the features. For example, chain codes can be used to represent curves, and quadtrees can be used to store an image with a large number of regions [33]. This approach requires extensive preprocessing, like fitting curves to contours. Scaling or expansion can then be easily accomplished since all the information has been translated to a scale-independent form.

Some work has been done to convert low resolution binary image data for high resolution displays [37]. The motivation lies in the reproduction of text data, obtained from "coarse scanning" devices, using higher quality printers and enlargement techniques for binary images. Techniques include logical expansion; a pixel value is based in the outcome of some predicates involving the logical values of the neighboring pixels. For example, the decision to fill in a pixel may depend on the number of painted neighboring pixels according to some majority rule: a pixel is painted if there are more white pixels than black ones. Similarly, one can employ some directional rule in which a pixel is painted if it is in the path of some painted pixels. By selecting a set of pixel configurations as paths, one can also perform logical smoothing [9, 27]. Other schemes may employ lookup tables to fill in the pixels by matching templates with neighborhoods around pixels [40].

In image processing, the topic of resolution conversion has been pursued indirectly in signal sampling issues. The sampling process can be approached in either the spatial or frequency domain. It is, however, more convenient to discuss it in the latter domain. When a one-dimensional signal is sampled by multiplying the signal function by a train of impulses (Dirac delta functions), the Fourier transform of the signal function, which is called the spectrum of the signal, is replicated along the frequency variable axis. The period of this replication is exactly the reciprocal of the interval length between the sampling pulses in the spatial domain. Therefore, in order to reproduce the original signal from the samples, it is only

necessary to remove the replicas except the primary one located about the origin. This can be done in the frequency domain by multiplying the "replica" function with a discontinuous function which is unity between a fixed interval about the origin and zero everywhere else. It turns out that this multiplication takes the form of a convolution of the sampled signal with a *sinc* function which is of the form $\frac{\sin x}{x}$ in the spatial domain. Hence, the original signal can be fully recovered by convolving the sampled impulses with a *sinc* function. One can then conclude that from the Sampling Theorem that the *sinc* function allow perfect interpolation. However, there are many difficulties in doing this. Questions like:

1. Are the images band-limited, i.e., the spectrum is non-zero only in a finite frequency range ?
2. Is there a critical sampling interval to enable reconstruction ?
3. If there are limitations to the accuracy of the equipment for sampling, what mathematical tools are available for reconstruction and how well do these methods rate ?

These questions have been extensively researched. Hou and Andrews [18] have demonstrated the use of B-Splines for interpolation and studied their performance by examining how well the Fourier transform of the cubic B-Spline clip the replicas in the spectrum of the sampled image. Pratt [30] has presented a number of interpolating filters, i.e., the Fourier transforms of interpolating functions, and discussed their interpolating errors.

In this thesis, the terms "resolution conversion" and "image expansion" will be treated synonymously. This stems from the possibility of spatially mapping the pixels of an appropriately expanded image onto the cells of a given higher resolution device to allow for the display of low resolution data on higher resolution devices.

(Digital scaling is the process performed on a digital input image, resulting in a digital output image of a different size at a fixed resolution.) Thus, in this thesis, attention is focussed on taking a set of multi-level data and producing another set, which represents the original image at a desired scale by interpolation or using spatial noise-cleaning techniques to smooth out blocked artifacts of expanded images obtained by pixel replication. Although the methods are also applicable to binary images by thresholding the processed multi-level image, these schemes have not necessarily exploited the bilevel nature of the images to the fullest. Moreover, the question of choosing proper thresholds needs to be investigated. Therefore, although the methods discussed in this thesis may not be limited to multi-level images, the performance may not be suitable for binary images.

1.4. Thesis Organization

In the next chapter, the scaling of chain codes for expanding lineal features in an image is briefly reviewed. Interpolation is discussed from both the viewpoint of the spatial and frequency domains. Finally, the bilinear and various piecewise bicubic schemes are developed. Noise-cleaning techniques are surveyed in Chapter 3. Chapter 4 reviews some of the most common edge enhancement techniques. The results of some of the methods from Chapter 2, 3 and 4 are discussed in Chapter 5. Chapter 6 investigates the validity of the application of the results and extensions to the problem.

Chapter 2

Expansions

When an object in an image is described by a set of vectors, expansion of this object is made by multiplying the end points of the vectors by the scaling factor and setting pixels that lie on straight lines between appropriate pairs of points to the original gray values. If the contents of an image are not known, however, (except for the gray value at each pixel) the above scheme is not immediately applicable. For such a case, the image can be expanded by scaling the distance between pixels and interpolating the gray values in the gaps between the scaled pixels.

In this chapter, various interpolation methods are examined. Images which contain curves and well defined regions can be expanded effectively if there are descriptions of these entities. Expansion techniques which exploit this information are treated in Section 2.1. The interpolation methods and their problems are discussed in Sections 2.2 and 2.3.

2.1. Scaling

The treatment of image expansion in this section assumes that the entities in the image are known and their descriptions are available; for example, the chain codes for the curves or contours of regions are accessible. A curve can be expanded along its arc length by replicating each link in the chain code by the expansion factor. Similarly, regions can be expanded equally in both horizontal and vertical directions by expanding the contours and filling in the pixels in the regions. Evidently, this assumes that gray values of the pixels on the curve or contour are

identical. In many cases, this is not true and a gap filling algorithm is required; linear interpolation can be applied to the curve and bilinear interpolation for the interior pixels of regions. The interpolation schemes discussed in Section 2.3 may be used.

If, however, the expansion is not equal in both horizontal and vertical directions, the above technique becomes cumbersome; the diagonal links can no longer be replicated like the vertical and horizontal counterparts. Instead, the values beside the diagonal links must be determined by the proportion of the expansion in either axial directions and the direction of the curve. For example, the curve 1112 when expanded twice in the x direction may take the form of either 1010102 or 0101012. The non-uniqueness of the expanded curve arises from the discreteness of the image plane.

When the endpoints of the curves or contours are described in rectangular coordinates, they can be scaled by S_x along the x -axis and by S_y in the y -axis into new points by multiplying the abscissae by S_x and ordinates by S_y . In this case, the pixels that lie in the straight line between expanded points are determined and given the proper gray values. As seen from the previous example, such a straight line is not necessarily unique. For gray-scaled images, techniques for smoothing jaggies can be used to provide a more pleasant straight line; the subarea or percentage of the pixel area occupied by the crossing line is evaluated and a proportional gray value is given to each pixel in the path of the straight line [6]. Moreover, this expansion is relative to the origin.

In practice, the objects in an image are not always well-defined and known. Very often, the images are used to locate objects and it would be difficult to apply the above techniques. Features like bodies of water or roads in an aerial terrain image may not be well-defined as a consequence of atmospheric corruption or other

effects. Thus, the application of above techniques necessitates the initial identification of these features by matching and edge detection techniques.

In the next two sections, interpolation which offers a more general approach is investigated.

2.2. Interpolation in the Frequency Domain

In order to discuss the role of interpolation in the frequency domain, it is necessary to examine the problems connected to sampling, including those alluded to in Chapter 1. Before investigating the ramifications of sampling continuous tone images and of processing the sampled data, a brief review of the Fourier transform is made. Also, for convenience, one dimensional signal functions are considered.

2.2.1. Preliminary Definitions and Notations

The following are some functions that will be used in the course of discussing the Fourier transform. The definitions correspond to those in [2].

The rectangular function of unit base and height is defined as:

$$R(t) = \begin{cases} 1 & \text{if } |t| \leq \frac{1}{2} \\ 0 & \text{if } |t| > \frac{1}{2} \end{cases} \quad (2.1)$$

Therefore, $hR(\frac{(t-c)}{b})$ is a displaced rectangle function of height h and base b , centered at $t=c$.

The triangle function which is also called the *Chateau* function, is defined as:

$$T(t) = \begin{cases} 0 & \text{if } |t| \leq 1 \\ 1-|t| & \text{if } |t| > 1 \end{cases} \quad (2.2)$$

Again, $hT(\frac{(t-c)}{b})$ is a displaced triangle function centered at c with a height of h and base $2b$.

The *impulse* or *Dirac delta* function is not a function by traditional definition. It is a symbolic function defined by its integral property.

$$\int_{-\infty}^{\infty} \delta(t) dt = \int_{-\epsilon}^{\epsilon} \delta(t) dt = 1, \quad (2.3)$$

where ϵ is an arbitrary small magnitude. A consequence of (2.3) is

$$\int_{-\infty}^{\infty} f(t) \delta(t) dt = f(0), \text{ and} \quad (2.4)$$

$$\int_{-\infty}^{\infty} f(t) \delta(t - t_0) dt = f(t_0). \quad (2.5)$$

The step function is a function discontinuous at $t=0$. It is given by

$$U(t) = \begin{cases} 1 & \text{if } t > 0 \\ 0 & \text{if } t \leq 0. \end{cases} \quad (2.6)$$

The *shah* function $III(t)$ is an infinite impulse train and is defined as

$$III(t) = \sum_{n=-\infty}^{\infty} \delta(t-n). \quad (2.7)$$

The convolution integral, in one dimension, of two functions $f(t)$ and $g(t)$ is

$$\int_{-\infty}^{\infty} f(u) g(t-u) du, \quad (2.8)$$

or simply,

$$f(t) * g(t). \quad (2.9)$$

The convolution operation has several properties:

$$\text{commutativity: } f * g = g * f, \quad (2.10)$$

$$\text{distributive over addition: } f * (g + h) = f * g + f * h, \quad (2.11)$$

$$\text{associativity: } f * (g * h) = (f * g) * h. \quad (2.12)$$

The two dimensional definition of convolution extends in the obvious manner:

$$\begin{aligned} h(x, y) &= f * g \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(u, v) g(x-u, y-v) du dv. \end{aligned} \quad (2.13)$$

Similarly, the discrete versions of (2.8) and (2.13) follow naturally as

$$\begin{aligned} h(i) &= f^*g \\ &= \sum_m f(m)g(i-m) \end{aligned} \quad (2.14)$$

and

$$\begin{aligned} h(i,j) &= f^*g \\ &= \sum_m \sum_n f(m,n)g(i-m,j-n) \end{aligned} \quad (2.15)$$

where the summations are taken only over the area of non-zero overlap.

2.2.2. Definitions, Properties and Useful Results of the Fourier Transform

The Fourier transform of a one-dimensional function $f(t)$ is defined [2] as

$$\begin{aligned} \mathbf{F}\{f(t)\} &= F(s) \\ &= \int_{-\infty}^{\infty} f(t)e^{-i2\pi st} dt. \end{aligned} \quad (2.16)$$

The inverse Fourier transform of $F(s)$ is defined as

$$\mathbf{F}^{-1}\{F(s)\} = \int_{-\infty}^{\infty} F(s)e^{i2\pi st} ds. \quad (2.17)$$

According to the Shifting Theorem [2]

$$\mathbf{F}\{f(t-a)\} = e^{-i2\pi as}\mathbf{F}\{f(t)\}. \quad (2.18)$$

The Similarity Theorem [2] states that if $\mathbf{F}\{f(t)\} = F(s)$ then

$$\mathbf{F}\{f(at)\} = \frac{1}{|a|}F\left(\frac{s}{a}\right). \quad (2.19)$$

For any $f(t)$, the Fourier transform $F(s)$, or the spectrum, is unique and vice-versa.

One of the problems that needs to be addressed is the existence of the integrals in the transform equations. In reality, most functions satisfy the criteria for existence and the existence problem is not considered as a major problem. (See [9] for a more detailed treatment.)

Table 2.1 lists the Fourier transforms of some common functions [2].

Table 2.1 Fourier Transforms of Some Functions.:

Name	Signal $f(t)$	Spectrum $F(s)$
Function	$f(t)$	$F(s)$
Gaussian	$e^{-\pi t^2}$	$e^{-\pi s^2}$
Rectangular Pulse	$R(t)$	$\frac{\sin(\pi s)}{\pi s}$
Sinc	$\frac{\sin(\pi t)}{\pi t}$	$R(s)$
Triangular Pulse	$T(t)$	$\frac{\sin^2(\pi s)}{(\pi s)^2}$
Sinc ²	$\frac{\sin^2(\pi t)}{(\pi s)^2}$	$T(s)$
Impulse	$\delta(t)$	1
Unit Step	$U(t)$	$\frac{1}{2}[\delta(s) - \frac{i}{\pi s}]$
Shah	$III(t)$	$III(s)$
Cosine	$\cos(2\pi ft)$	$\frac{1}{2}[\delta(s+f) + \delta(s-f)]$
Sine	$\sin(2\pi ft)$	$\frac{i}{2}[\delta(s+f) - \delta(s-f)]$
Complex Exponential	$e^{i2\pi ft}$	$\delta(s-f)$

The most important theorem for linear system analysis is the convolution theorem:

$$\mathbf{F}\{f(t) * g(t)\} = F(s)G(s), \quad (2.20)$$

$$f(t)g(t) = \mathbf{F}^{-1}\{F(s) * G(s)\} \quad (2.21)$$

where $F(s)$ and $G(s)$ are the Fourier transforms of $f(t)$ and $g(t)$, respectively. The theorem relates two operations, one in the spatial domain and the other in the frequency domain. One implication is that an expensive operation in one domain can be performed more economically in the other for the same effect.

The total energy content E of a function $f(t)$ is defined as

$$E = \int_{-\infty}^{\infty} |f(t)|^2 dt, \quad (2.22)$$

provided the integral exists.

Rayleigh's theorem [2] states that

$$\int_{-\infty}^{\infty} |f(t)|^2 dt = \int_{-\infty}^{\infty} |F(s)|^2 ds, \quad (2.23)$$

thus implying the conservation of energy when a function is transformed from the spatial domain to the frequency domain.

The autocorrelation is the convolution of a function with itself but without reflection:

$$\begin{aligned} P_f(\tau) &= f(t) * f(-t) \\ &= \int_{-\infty}^{\infty} f(t)f(t+\tau)dt. \end{aligned} \quad (2.24)$$

The autocorrelation is always even and has a maximum at $\tau=0$. It has the property

$$\int_{-\infty}^{\infty} P_f(\tau)d\tau = \left[\int_{-\infty}^{\infty} f(t)dt \right]^2. \quad (2.25)$$

The Fourier transform of the autocorrelation function given by

$$\begin{aligned} P_f(s) &= F\{f(t) * f(-t)\} \\ &= |F(s)|^2, \end{aligned} \quad (2.26)$$

is called the power spectrum of $f(t)$. The cross correlation function which indicates the amount of agreement between two functions $f(t)$ and $g(t)$ for various degrees of misalignment is defined as

$$P_{fg}(\tau) = f(t) * g(-t). \quad (2.27)$$

The cross-power spectral density function is the Fourier transform of the cross-correlation function and therefore is

$$P_{fg}(s) = F\{f(t) * g(-t)\}. \quad (2.28)$$

2.2.3. Sampling

In this section, the theory behind a sampling technique is reviewed. This is to facilitate the discussion of the cause of aliasing and the validity of interpolation filters. Most image processing problems are modeled as linear shift invariant systems. Fig. 2.1 shows the terminology commonly used for a linear system.

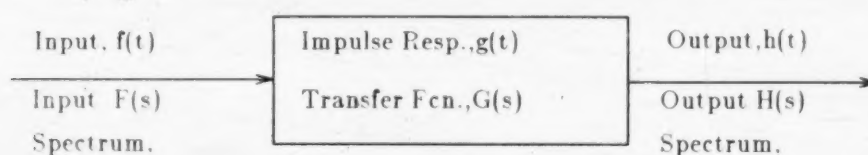


Figure 2.1 Terminology of a Linear System.

In general, the Fourier transform of the impulse response $g(t)$ is called the transfer function $G(s)$. $F(s)$ and $H(s)$ are referred to as the spectrum of the input and output signals, respectively. The convolution theorem (2.20), (2.21) allows such a system to be specified completely by either the impulse response function or the transfer function.

A function $f(t)$ is band-limited at a frequency s_0 if its spectrum is zero in the intervals $(-\infty, -s_0]$ and $[s_0, \infty)$. That is,

$$F(s) = 0, |s| \geq s_0. \quad (2.29)$$

The process of sampling $f(t)$ at equal intervals of τ can be modeled by multiplying $f(t)$ with $\frac{1}{\tau} \text{III}(\frac{t}{\tau})$. This will zero out $f(t)$ between the sampling points $n\tau$ with $n = 0, \pm 1, \pm 2, \dots$, and preserve the value of $f(t)$ at the sample points in the strength of the resulting impulses [3].

According to the convolution theorem, multiplying $f(t)$ with $\frac{1}{\tau} \text{III}(\frac{t}{\tau})$ convolves the spectrum $F(s)$ with $\text{III}(\tau s)$. Since $\text{III}(\tau s)$ is a train of impulses of

strength $\frac{1}{T}$ at equally spaced intervals of $\frac{1}{T}$ along the frequency variable axis, i.e., s -axis, performing the convolution would then replicate the spectrum $F(s)$ at every $\frac{1}{T}$ intervals along the s -axis from minus infinity to plus infinity. The spectrum $H(s)$ of the sampled signal $h(t)$ consists of $F(s)$ spaced every $\frac{1}{T}$ apart.

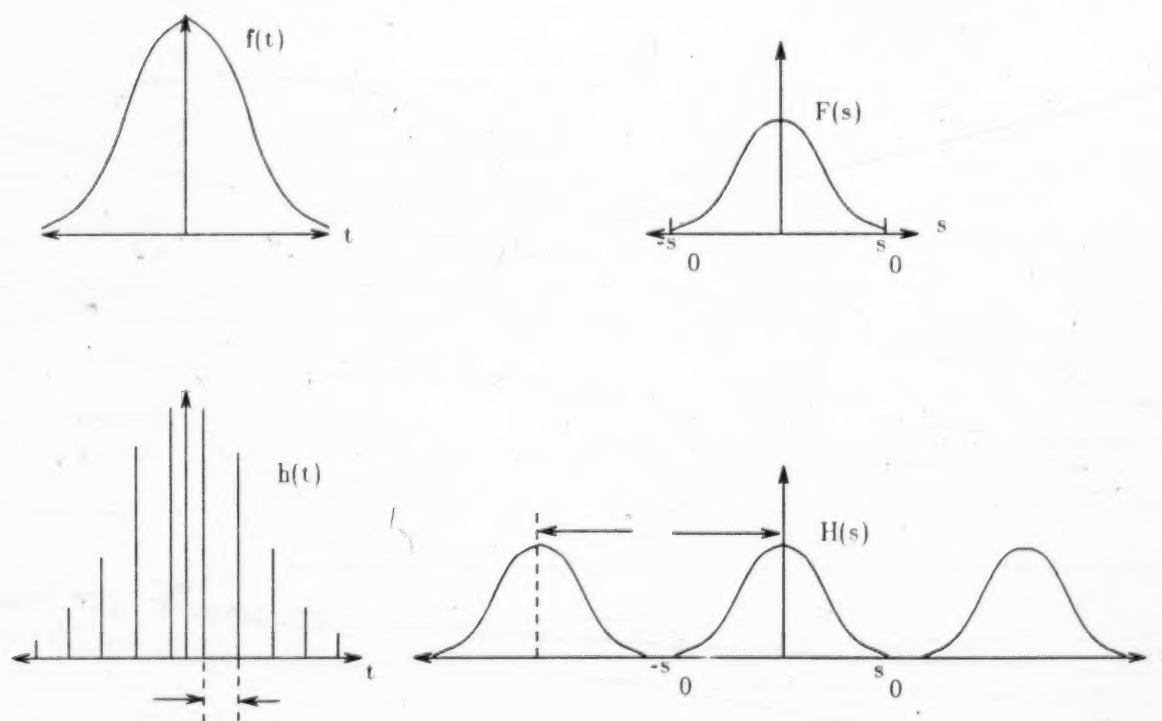


Figure 2.2 Sampling: Spatial and Freq. Domains.

Thus, in order to recover the original signal, either $h(t)$ is interpolated for $f(t)$ in the spatial domain, i.e., along the t -axis or the copies of $F(s)$, except for the primary one located about the origin, are eliminated in $H(s)$ to recover $F(s)$.

In this section the latter scheme is examined. One way to zero out the replicas of $F(s)$ is to multiply $H(s)$ with the rectangle function $R(\frac{s}{2s_1})$ which is unity in the

interval $[-s_1, s_1]$ and zero everywhere else. The value s_1 ranges from s_0 to $\frac{1}{\tau} - s_0$ since the smallest frequency that can be used to recover $F(s)$ without clipping any frequency in the main cycle is s_0 . On the other hand, s_1 is not restricted until $\frac{1}{\tau} - s_0$ where the next replica occurs [3]. Thus, with

$$h(t) = \frac{1}{\tau} f(t) \text{III}\left(\frac{t}{\tau}\right), \quad (2.30)$$

or

$$H(s) = F(s) * \text{III}(\tau s), \quad (2.31)$$

the original signal is obtained by

$$F(s) = H(s) R\left(\frac{s}{2s_1}\right). \quad (2.32)$$

The transfer function $R\left(\frac{s}{2s_1}\right)$ is termed an interpolating filter. Its impulse response as seen in Table 2.1 is $\frac{2s_1 \sin(2\pi s_1 t)}{2\pi s_1 t}$. By convention, $\frac{\sin(\pi x)}{\pi x}$ is written as $\text{sinc}(x)$.

Again, appealing to the convolution theorem,

$$\begin{aligned} f(t) &= h(t) * 2s_1 \text{sinc}(s_1 t) \\ &= 2s_1 \sum_{m=-\infty}^{\infty} h(m) \text{sinc}(s_1(t-m)). \end{aligned} \quad (2.33)$$

Therefore, by using $\text{sinc}(t-m)$ as the basis function in an interpolation scheme, it is possible to perfectly reconstruct the signal $f(t)$. This form of interpolation is often referred to as Cardinal Spline interpolation [18]. Hou et al. has also observed several problems with this method:

1. The Cardinal Spline behaves like an infinite degree polynomial whose support is not local and poses a computational problem. If truncations are made in the lower and upper bounds of the summation in (2.33), oscillation known as the Gibb's phenomenon will show up in the interpolated signal.

2. Most images are not band-limited.

Taking these into consideration, it is obvious that the possibility of a perfect reconstruction by interpolation in the spatial domain is remote. Approximations like the Gaussian [30] filter, cubic B-Spline [18] have been used. In this thesis, the application of piecewise Hermite interpolation, natural and complete splines are studied. Although it is difficult to reconstruct $f(t)$ perfectly in the spatial domain, it is apparent from (2.32) that $f(t)$ can be recovered by performing the inverse transform on $H(s)R(\frac{s}{2s_1})$. It shall be shown, however, that this is difficult in digital operations.

When $f(t)$ is sampled, a continuous periodic spectrum $F(s)$ is obtained. But in reality, when the Fourier transform of $\frac{1}{T}f(t)III(\frac{t}{T})$ is calculated, only the primary cycle of $F(s)$ is calculated. Moreover, it is calculated at equally spaced points. Thus, in fact, another spectrum $\hat{F}(s)$ is produced by sampling $F(s)$ at equal intervals. Since $\hat{F}(s)$ is sampled, its inverse transform $\hat{f}(t)$ is a continuous periodic function of infinite duration. Therefore, $\hat{f}(t)$ is not $f(t)$ or even $\frac{1}{T}f(t)III(\frac{t}{T})$, but rather a periodic function consisting of copies of $f(t)$.

Suppose $\hat{F}(s)$ is multiplied to $R(\frac{s}{2s_1})$ to produce the primary cycle. This product in the frequency domain corresponds to convolving $\text{sinc}(s_1 t)$ with $\hat{f}(t)$. Since $\hat{f}(t)$ is periodic and $\text{sinc}(s_1 t)$ extends from negative infinity to infinity, the convolution will result in smearing between adjacent cycles of $\hat{f}(t)$. And if the resulting function is windowed about the origin in an attempt to reclaim $f(t)$, high frequencies introduced by the smearing at the tail ends of $\hat{f}(t)$ will appear as artifactual discontinuities. These difficulties justify the pursuit of other interpolating methods to recover or approximate the original signal from its

samples.

Aliasing, as was mentioned in Section 1.2, is a result of folding of energy in the primary cycle of the spectrum of the signal. Referring to Fig. 2.2 and (2.29), (2.30), it is evident that the replicas will overlap if the signal is not band-limited or the sampling rate is not high enough. Evidently from Fig. 2.2, the intervals between the replicas must be at least twice the band-width, s_0 , i.e.,

$$\frac{1}{\tau} \geq 2s_0, \quad (2.34a)$$

or

$$\tau \leq \frac{1}{2s_0} \quad (2.34b)$$

where the signal is band-limited at s_0 and τ is the sampling rate of the signal. In the rest of this thesis, adequate sampling is assumed.

In the next section, interpolation methods for the gray levels between pixels are examined as an approach to achieving expansion and maintaining continuity of the contents of the images, when the contents of the images are not known, except for their gray levels.

2.3. Interpolation in the Spatial Domain

From the numerical analysis point of view, it is known that an increasing set of points can be interpolated by using polynomials of higher degree. The problem with high degree polynomials is its notorious tendency to oscillate so that pinning down a polynomial at a few points for a slowly varying function may not produce a good approximation of the function [18]. Also, since they are analytic, their global behavior is determined by the behavior in any interval; a change in the functional behavior in an interval is felt everywhere. These problems motivate the employment of piecewise interpolation. Literature on piecewise interpolation is

abundant [8,36].

In this section, the assumption of a priori knowledge of an image is dropped. Here, an image is expanded by scaling the distance between pixels and polynomial interpolation is considered to estimate the gray values between the expanded pixels. The simplest method is bilinear interpolation, which is discussed in Section 2.3.1. Higher order interpolating schemes generally yield better appearing results. Section 2.3.3 discusses the development of bicubic spline interpolation along the lines of the derivation of Hermite bicubic interpolation presented in [12].

2.3.1. Bilinear Interpolation

Suppose an image is defined to be an m by n array of gray values, $f(x, y)$, $0 \leq x < m$, $0 \leq y < n$. The expanded image $g(x, y)$, $0 \leq x < (m-1)S_x$, $0 \leq y < (n-1)S_y$ is computed as follows:

$$\begin{aligned} g(iS_x + k, jS_y + l) = & \left(\frac{S_x - k}{S_x} \right) \left(\frac{S_y - l}{S_y} \right) f(i, j) \\ & + \left(\frac{S_x - k}{S_x} \right) \frac{l}{S_y} f(i, j+1) \\ & + \frac{k}{S_x} \left(\frac{S_y - l}{S_y} \right) f(i+1, j) \\ & + \frac{kl}{S_x S_y} f(i+1, j+1), \end{aligned} \quad (2.35)$$

for $0 \leq i < m-1$, $0 \leq j < n-1$, $0 \leq k < S_x$, $0 \leq l < S_y$.

In this method, the expanded image is divided into mn patches, each of which is S_x by S_y . The gray levels of the pixels in the patch $\{(x, y) \mid x = iS_x, iS_x+1, \dots, iS_x+S_x-1, y = jS_y, jS_y+1, \dots, jS_y+S_y-1\}$ for $0 \leq i \leq m-1$ and $0 \leq j \leq n-1$, are interpolated bilinearly.

It should be noted that interpolation can be performed using linear interpolation in the x direction followed by the y direction (or vice versa). In Fig.

2.3, the gray value at E can be found by interpolating first for A and C (or B and D) and followed by interpolating between them.

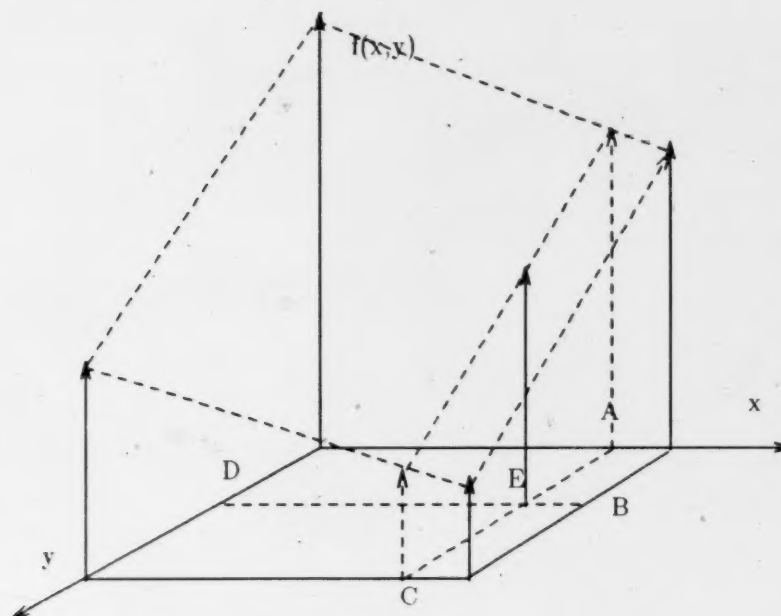


Figure 2.3 Bilinear Interpolation

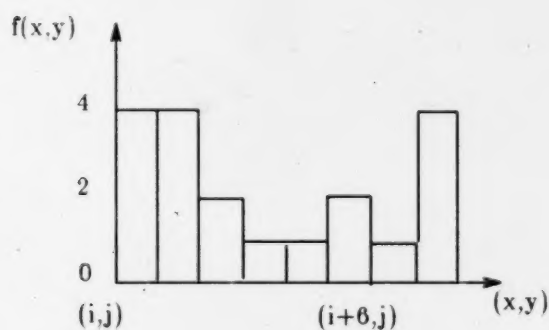


Figure 2.4 Replication of Pixels

The interpolation between A and C is a weighted average of the intensity levels at A and C with the distances between the point of interpolation and A, C as the

relative weights. Similarly, the distances between E and (iS_x, jS_y) , $((i+1)S_x, jS_y)$, $(iS_x, (j+1)S_y)$ and $((i+1)S_x, (j+1)S_y)$ act as the weights in the bilinear interpolation at E where E is the pixel at $(iS_x + k, jS_y + l)$, $0 \leq k < S_x$, $0 \leq l < S_y$. This is illustrated in Fig. 2.3.

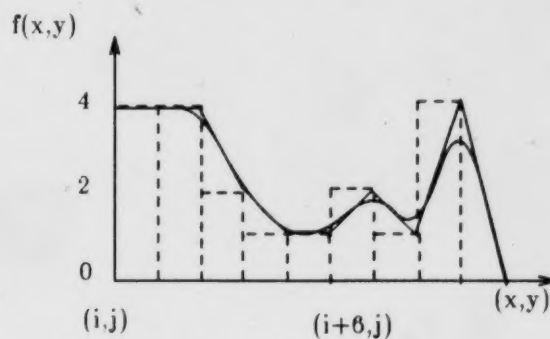


Figure 2.5 (Bi) Linear Interpolation

However, this simple approach will usually lead to blurred edges, since the pixels lying between the original interior edge pixel and exterior edge pixel take on a weighted average of these two pixels. To see this, consider Figures 2.4 and 2.5 which depict the gray levels as a function of the location of the pixels of an expanded image. In this case, pixels (iS_x, jS_y) , $((i+1)S_x, jS_y)$, $\dots$, $((i+9)S_x, jS_y)$ are considered.

In the original image, the gray values at (i,j) , $(i+1,j)$, $\dots$, $(i+9,j)$ are 4, 4, 4, 2, 1, 1, 2, 1, 4, 0, respectively. Fig. 2.4 illustrates the case when an expansion is performed by straight replication of pixels (zooming). The blocked appearance is due to the discontinuity of the curve. Bilinear interpolation as seen in Fig. 2.5, reduces the blocky appearance but Mach bands, caused by the discontinuities of the first derivative (slope), manifest themselves at positions like $((i+5)S_x, jS_y)$ and $((i+6)S_x, jS_y)$. Since interpolation is done piecewise in patches of S_x by S_y pixels,

these Mach bands cause the image to be slightly blocky [12].

Clearly from pixels $((i+8)S_x, jS_y)$ and $((i+9)S_x, jS_y)$ of Fig. 2.5, the extent of the edges being blurred depends on the expansion factors, S_x and S_y and the difference in the adjacent gray levels.

By using higher order interpolation, the gray level function is at least C^1 , i.e., the first derivative is continuous and the presence of Mach bands is reduced. These effects can be improved by requiring the interpolating functions to change rapidly when there is a large change in gray values of adjacent pixels (in the original image) as indicated by the smoother line in Fig. 2.5. A common choice for higher interpolating functions are the cubic polynomials. Unlike higher degree polynomials, oscillations are controlled and no lower order representation of curve segments provides continuity of the curve and the slope like cubic functions. Cubic interpolation is discussed next.

2.3.2. Piecewise Cubic Interpolation

The gray levels of an image as a function of their positions can be looked upon as a surface so that an interpolation of the surface corresponds to interpolating the gray levels. The interpolation is called *piecewise* when the surface is divided into rectangles and the gray values of pixels in each rectangle are obtained by interpolating gray values of certain pixels. In bilinear interpolation, the surface is interpolated in patches of S_x and S_y using the four corner points of each patch in the manner described in the previous section. Each interpolation can be obtained in two steps by successively interpolating along the rows and columns (or vice-versa) with linear interpolation. Higher order surface interpolation can be sought analogously. For example, bicubic interpolation can be found by interpolating using cubic polynomials along the x and y direction in each patch.

In this section, some piecewise cubic interpolating polynomials, namely the Hermite, Bezier, complete and natural splines, are briefly reviewed. The interpolation problem is then extended to two dimensions in the next section.

In keeping the notation consistent with bilinear interpolation, it is assumed that an image $f(x, y)$, $0 \leq x < m$ and $0 \leq y < n$, is expanded by S_x and S_y in the x and y directions, respectively so that the pixels (i, j) are mapped to (iS_x, jS_y) , $0 \leq i < m$, $0 \leq j < n$. Given the gray values, $f(0, j'S_y), f(S_x, j'S_y), \dots, f((m-1)S_x, j'S_y)$ for a fixed j' , $0 \leq j' < n$, a piecewise cubic interpolant g to f is constructed. For convenience, the gray values $f(iS_x, j'S_y)$, $0 \leq i < m$ is written as $f(i)$. On each interval $[(iS_x, j'S_y), ((i+1)S_x, j'S_y)]$ (or $[i, i+1]$), g agrees with some cubic polynomial P_i , $0 \leq i < m-1$. That is,

$$\begin{aligned} g(x, j'S_y) &= P_i(x - iS_x, j'S_y), \\ &\text{for } iS_x \leq x \leq (i+1)S_x, \end{aligned} \quad (2.36)$$

and some cubic polynomial P_i , $i = 0, \dots, m-2$ or by a similar change in notation for convenience,

$$\begin{aligned} g(x) &= P_i(x - i) \\ &\text{for } i \leq x \leq i+1, i = 0, \dots, m-2. \end{aligned} \quad (2.37)$$

The i^{th} polynomial P_i is made to satisfy the conditions:

$$P_i(0) = f(i) \quad (2.38a)$$

$$P_i(1) = f(i+1) \quad (2.38b)$$

$$P_i'(0) = d_i \quad (2.38c)$$

$$P_i'(1) = d_{i+1} \quad (2.38d)$$

for $i = 0, 1, \dots, m-2$, where d_i and d_{i+1} are free parameters. Also, the resulting piecewise cubic function g agrees with f at $0, 1, \dots, m-1$ and has a continuous first derivative on $[0, m-1]$ regardless of how the free "slopes", d_i 's are selected.

Assume that

$$P_i(x) = a_{1,i}x^3 + a_{2,i}x^2 + a_{3,i}x + a_{4,i} \quad (2.39a)$$

for $x \in [0,1]$, and $i = 0, \dots, m-2$.

This can be expressed as a dot product:

$$P_i(x) = [x^3 \ x^2 \ x \ 1] \begin{bmatrix} a_{1,i} \\ a_{2,i} \\ a_{3,i} \\ a_{4,i} \end{bmatrix} \quad (2.39b)$$

$$= XA_i \quad (2.40)$$

From equations (2.38) and (2.39b),

$$\begin{bmatrix} f(i) \\ f(i+1) \\ d_i \\ d_{i+1} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \begin{bmatrix} a_{1,i} \\ a_{2,i} \\ a_{3,i} \\ a_{4,i} \end{bmatrix} \quad (2.41)$$

for $i = 0, 1, \dots, m-2$. By inverting the 4 by 4 matrix on the r.h.s. of (2.41),

$$\begin{bmatrix} a_{1,i} \\ a_{2,i} \\ a_{3,i} \\ a_{4,i} \end{bmatrix} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} f(i) \\ f(i+1) \\ d_i \\ d_{i+1} \end{bmatrix} \quad (2.42a)$$

for $i = 0, 1, \dots, m-2$,

$$\text{or } A_i = M_h G \quad (2.42b)$$

where M_h is the 4 by 4 matrix on the r.h.s. of (2.42) and is called the Hermite matrix, and G is the column vector $[f(i), f(i+1), d_i, d_{i+1}]^T$ [12].

Substitution of (2.42) into (2.40) gives

$$P_i(x) = XM_h G \quad (2.43)$$

The cubic Hermite interpolation is obtained by choosing the parameters

d_i , $0 \leq i < m-1$ to be $g'(i)$. Therefore, from (2.43),

$$P_i(x) = XM_h G_h \quad (2.44)$$

$$\text{where } G_h = \begin{bmatrix} f(i) \\ f(i+1) \\ g'(i) \\ g'(i+1) \end{bmatrix}$$

is called the Hermite geometry vector, and $0 \leq i < m-1$.

The Bezier interpolation resembles the cubic Hermite by requiring the parameter, d_i to be the slope at i . These guiding points are usually specified interactively by the user in applications where cubic curves are generated [12].

The tangent vector at i is, however, defined to be proportional to the difference between $f(i)$ and a guiding point $h(i)$. Specifically,

$$d_i = 3(h(i) - f(i)) \quad (2.45a)$$

$$d_{i+1} = 3(f(i+1) - h(i+1)) \quad (2.45b)$$

where $0 \leq i < m-1$.

Hence, using (2.43),

$$\begin{bmatrix} a_{1,i} \\ a_{2,i} \\ a_{3,i} \\ a_{4,i} \end{bmatrix} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} f(i) \\ f(i+1) \\ 3(h(i) - f(i)) \\ 3(f(i+1) - h(i+1)) \end{bmatrix} \quad (2.46)$$

$$= \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -3 & 3 & 0 & 0 \\ 0 & 0 & -3 & 3 \end{bmatrix} \begin{bmatrix} f(i) \\ h(i) \\ h(i+1) \\ f(i+1) \end{bmatrix} \quad (2.47)$$

$$= M_h M_{hb} G_b \quad (2.48)$$

for $i = 0, 1, \dots, m-2$. G_b , the Bezier geometry vector, is thus related to the Hermite geometry vector G_h by

$$G_h = M_{hb} G_b \quad (2.49a)$$

It follows from either (2.43) or (2.44) that

$$P_i(x) = XM_h M_{hb} G_b. \quad (2.49b)$$

While the Hermite and Bezier forms have zero and first order continuities, (interpolating) splines are smoother in the sense that they also have second order continuity. The parameters $d_1, \dots, d_{m-2}$ are determined from the conditions that the interpolant g should be twice continuously differentiable. Therefore, for $i = 1, 2, \dots, m-2$,

$$P''_{i-1}(1) = P''_i(0) \quad (2.50)$$

or by differentiating (2.39a) twice, (2.50) implies

$$6a_{1,i-1} + 2a_{2,i-1} = 2a_{2,i}. \quad (2.51)$$

From (2.41), (2.51) translates to

$$3(f(i+1) - f(i-1)) = d_{i-1} + 4d_i + d_{i+1} \quad (2.52)$$

for $i = 1, 2, \dots, m-2$.

Assuming that the two remaining parameters d_0 and d_{m-1} have been chosen, then (2.52) is the tridiagonal linear equation system of $m-2$ equations for the $m-2$ unknowns $d_1, \dots, d_{m-2}$:

$$\begin{bmatrix} 4 & 1 & & & \\ 1 & 4 & 1 & & \\ & 1 & 4 & 1 & \\ & & & & \\ & & & & \\ & & & 1 & 4 & 1 \\ & & & & 1 & 4 \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ \vdots \\ \vdots \\ d_{m-2} \end{bmatrix} = \begin{bmatrix} 3(f(2) - f(0)) - d_0 \\ 3(f(3) - f(1)) \\ 3(f(4) - f(2)) \\ \vdots \\ \vdots \\ 3(f(m-1) - f(m-3)) - d_{m-1} \end{bmatrix} \quad (2.53)$$

The coefficient matrix on the l.h.s. in (2.53) is tridiagonal, symmetric, diagonally

dominant with positive diagonal elements, and hence guarantees the existence and uniqueness of the interpolating spline if d_0 and d_{m-1} are given. This solution can be found using the Gaussian elimination. Various choices for d_0 and d_{m-1} give rise to different splines. The natural spline results from the free-end conditions

$$g''(0) = 0, \quad (2.54a)$$

$$g''(m-1) = 0. \quad (2.54b)$$

It follows from (2.39a) that

$$P_i''(0) = 2a_{2,i}, \text{ for } i = 0, 1, \dots, m-2, \quad (2.55a)$$

$$P_{m-2}''(1) = 6a_{1,m-2} + 2a_{2,m-2}. \quad (2.55b)$$

These second derivatives can be expressed in terms of the function values and derivatives using (2.42a) and (2.55). This will result in

$$P_i''(0) = 6[f(i+1) - f(i)] - 4d_i - 2d_{i+1} \quad (2.56a)$$

for $i = 0, 1, \dots, m-2$ and

$$P_{m-2}''(1) = 2[-3(f(m-1) - f(m-2)) + d_{m-2} + 2d_{m-1}]. \quad (2.56b)$$

Therefore, for the natural spline, in addition to solving the system (2.53), the equations

$$6[f(1) - f(0)] - 4d_0 - 2d_1 = 0, \text{ and} \quad (2.57a)$$

$$2[-3(f(m-1) - f(m-2)) + d_{m-2} + 2d_{m-1}] = 0 \quad (2.57b)$$

must be solved.

If $f'(0)$ and $f'(m-1)$ are known and are used as d_0 and d_{m-1} , respectively, the resulting spline interpolant is known as the complete cubic spline. Thus, when spline interpolants are used, (2.53) must be solved using the predetermined boundary conditions. The resulting values for $d_1, \dots, d_{m-2}$ are then used in (2.43).

The Hermite interpolation scheme requires the knowledge of the first derivatives at $i = 0, 1, \dots, m-2$. Unfortunately, the rate of change of intensities with respect to distance can only be estimated. Due to the discreteness of the

separation between pixels, there is considerable arbitrariness in determining the first derivatives. The first derivative of the gray levels at x can be estimated using first differences, i.e.,

$$\Delta_1 f(x) = f(x) - f(x-1). \quad (2.58)$$

It does not, however, seem correct since the difference is not symmetrical at x ; Δ_1 uses a pair of values centered at $x - \frac{1}{2}$. This can be avoided by using the more widely used formula:

$$\Delta_2 f(x) = \frac{1}{2}[(f(x+1) - f(x)) + (f(x) - f(x-1))] \quad (2.59a)$$

$$= \frac{1}{2}(f(x+1) - f(x-1)). \quad (2.59b)$$

Δ_2 is a "smoothing" operator, in the sense that the differences across $x+1$ and x , and x and $x-1$ are averaged.

The disadvantage to using $\Delta_2 f(x)$ is that the value $f(x)$ is not included. One way to allow for this inclusion is to define:

$$\begin{aligned} \Delta_3 f(x) &= \frac{1}{2}(\Delta_1(f(x+1)) - \Delta_1 f(x)) \\ &= \frac{1}{2}[(f(x+1) - f(x)) - (f(x) - f(x-1))] \\ &= \frac{1}{2}(f(x+1) + f(x-1) - 2f(x))^\dagger. \end{aligned} \quad (2.60)$$

2.3.3. Piecewise Bicubic Interpolation

A surface representing the gray levels of an image can be described by $f(x, y)$, $0 \leq x < m$, $0 \leq y < n$. As in the preceding section, the gray values at (iS_x, jS_y) , $0 \leq i < m$, $0 \leq j < n$ are known and denoted by $f(i, j)$ for convenience. The approach in defining the piecewise bicubic interpolants is analogous to the derivation of piecewise cubic functions. The piecewise interpolant g of f in the rectangle $\{(x, y) | i \leq x \leq i+1, j \leq y \leq j+1\}$ coincides with some bicubic function $B_{i,j}$.

[†] This finite difference approximates the second derivative more so than the first derivative.

$$g(x, y) = B_{i,j}(x-i, y-j) \text{ for } i \leq x \leq i+1, j \leq y \leq j+1, \quad (2.61)$$

and $0 \leq i < m-1, 0 \leq j < n-1$.

The form of $B_{i,j}(x, y)$ is

$$B_{i,j}(x, y) = a_{1,1}x^3y^3 + a_{1,2}x^3y^2 + a_{1,3}x^3y + a_{1,4}x^3 \quad (2.62)$$

$$+ a_{2,1}x^2y^3 + a_{2,2}x^2y^2 + a_{2,3}x^2y + a_{2,4}x^2$$

$$+ a_{3,1}xy^3 + a_{3,2}xy^2 + a_{3,3}xy + a_{3,4}x$$

$$+ a_{4,1}y^3 + a_{4,2}y^2 + a_{4,3}y + a_{4,4}$$

$$= [x^3 \ x^2 \ x \ 1] \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} \end{bmatrix} \begin{bmatrix} y^3 \\ y^2 \\ y \\ 1 \end{bmatrix} \quad (2.63)$$

$$= XA_{i,j}Y^T \quad (2.64)$$

for $0 \leq x \leq 1, 0 \leq y \leq 1$ and Y is $[y^3 \ y^2 \ y \ 1]$.

It should be noted that the $a_{k,l}$'s in (2.62) should have been written more precisely as $a_{k,l,i,j}$ since the $a_{k,l}$'s are different for each pair of i and j 's for $1 \leq k, l \leq 4$.

If one of the parameters is given a fixed value and the other parameter is varied from 0 to 1, a cubic curve is obtained. Hence (2.64) can be rewritten using (2.43) but with G as a function of y if the bicubic is interpolated in the y direction first.

$$B_{i,j}(x, y) = XM_h G(y) \quad (2.65)$$

$$= XM_h \begin{bmatrix} P_{i,j}(y) \\ P_{i+1,j}(y) \\ D_{i,j}(y) \\ D_{i+1,j}(y) \end{bmatrix} \quad (2.66)$$

for $0 \leq x, y \leq 1$ and $P_{i,j}, D_{i,j}$ are cubic interpolants of y where $i = 0, 1, \dots, m-2$

and $j = 0, 1, \dots, m-2$. $P_{i,j}(y)$ is the cubic polynomial that interpolates between (i, j) and $(i, j+1)$, and therefore, from (2.43)

$$P_{i,j}(y) = YM_h \begin{bmatrix} f(i, j) \\ f(i, j+1) \\ d_{i,j} \\ d_{i,j+1} \end{bmatrix} \quad (2.67)$$

for $0 \leq y \leq 1$, $i = 0, 1, \dots, m-2$ and $j = 0, 1, \dots, m-2$. The free parameters $d_{i,j}$ and $d_{i,j+1}$ in (2.67), and the free parameter functions $D_{i,j}$ and $D_{i+1,j}$ in (2.66) are different although they serve the same purpose as the d_i 's in (2.38). The cubic $D_{i,j}(y)$ interpolates for the x -direction free parameter at $(i, j+y)$ which is used to interpolate for $f(i+x, j+y)$ based on $f(i, j+y)$ and $f(i+1, j+y)$ whereas $d_{i,j}$ is the y direction free parameter at (i, j) and is used to interpolate for $f(i, j+y)$ using $f(i, j)$ and $f(i, j+1)$. Therefore, $D_{i,j}(y)$ can be expressed, using (2.43), as

$$D_{i,j}(y) = YM_h \begin{bmatrix} \partial_{i,j} \\ \partial_{i,j+1} \\ \delta_{i,j} \\ \delta_{i,j+1} \end{bmatrix} \quad (2.68)$$

for $0 \leq y \leq 1$, $i = 0, 1, \dots, m-1$ and $j = 0, 1, \dots, m-2$.

In (2.68), $\partial_{i,j}$ is the " x direction free parameter" at (i, j) and $\delta_{i,j}$ is the " y direction free parameter" at (i, j) of the " x direction free parameter" function $D_{i,j}$, $0 \leq y \leq 1$.

Hence,

$$\begin{bmatrix} P_{i,j}(y) \\ P_{i+1,j}(y) \\ D_{i,j}(y) \\ D_{i+1,j}(y) \end{bmatrix} = \begin{bmatrix} f(i, j) & f(i, j+1) & d_{i,j} & d_{i,j+1} \\ f(i+1, j) & f(i+1, j+1) & d_{i+1,j} & d_{i+1,j+1} \\ \partial_{i,j} & \partial_{i,j+1} & \delta_{i,j} & \delta_{i,j+1} \\ \partial_{i+1,j} & \partial_{i+1,j+1} & \delta_{i+1,j} & \delta_{i+1,j+1} \end{bmatrix} M_h^T Y^T \quad (2.69)$$

$$= Q_{i,j} M_h^T Y^T. \quad (2.70)$$

where M_h^T and Y^T are the transposes of M_h and Y .

Substituting (2.70) into (2.65) results in

$$B_{i,j}(x,y) = XM_h Q_{i,j} M_h^T Y^T \quad (2.71)$$

for $0 \leq x, y \leq 1$, $i = 0, 1, \dots, m-2$ and $j = 0, 1, \dots, m-2$.

According to (2.64) and (2.71), the coefficient matrix is

$$A_{i,j} = M_h Q_{i,j} M_h^T. \quad (2.72)$$

The bicubic Hermite patches are obtained by defining the free parameters as

follows: The "y direction free parameter" at (i,j) is the slope of the surface at (i,j) in the y direction, i.e.,

$$d_{i,j} = \left. \frac{\partial q(x,y)}{\partial y} \right|_{(i,j)} \quad (2.73)$$

for $i = 0, 1, \dots, m-1$ and $j = 0, 1, \dots, n-1$.

Similarly, the "x direction free parameter" at (i,j) is the slope of the surface at (i,j) in the x direction or

$$\partial_{i,j} = \left. \frac{\partial q(x,y)}{\partial x} \right|_{(i,j)} \quad (2.74)$$

for $i = 0, 1, \dots, m-1$ and $j = 0, 1, \dots, n-1$.

Finally, the "y direction free parameter" at (i,j) of the "x direction free parameter" function $D_{i,j}$ is the change of the x direction free parameter with respect to y, i.e.,

$$\begin{aligned} \delta_{i,j} &= \frac{\partial}{\partial y} \left(\left. \frac{\partial q(x,y)}{\partial x} \right|_{(i,j)} \right) \\ &= \left. \frac{\partial^2 q(x,y)}{\partial y \partial x} \right|_{(i,j)} \end{aligned} \quad (2.75)$$

for $i = 0, 1, \dots, m-1$ and $j = 0, 1, \dots, n-1$. The Hermite bicubic interpolation thus provides zero and first order continuities on the surface. The Bezier bicubic which shares these properties can be derived similarly. Here, as in the cubic

situation, the free parameters are defined exactly as the Hermite bicubic. However, guiding points are used to define the slopes:

$$d_{i,j} = 3(h_{1,2}(i,j) - f(i,j)) \quad (2.76a)$$

$$d_{i,j+1} = 3(f(i,j+1) - h_{1,3}(i,j)) \quad (2.76b)$$

$$d_{i+1,j} = 3(h_{4,2}(i,j) - f(i+1,j)) \quad (2.76c)$$

$$d_{i+1,j+1} = 3(f(i+1,j+1) - h_{4,3}(i,j)) \quad (2.76d)$$

$$\partial_{i,j} = 3(h_{2,1}(i,j) - f(i,j)) \quad (2.76e)$$

$$\partial_{i,j+1} = 3(h_{2,4}(i,j) - f(i,j+1)) \quad (2.76f)$$

$$\partial_{i+1,j} = 3(f(i+1,j) - h_{3,1}(i,j)) \quad (2.76g)$$

$$\partial_{i+1,j+1} = 3(f(i+1,j+1) - h_{3,4}(i,j)) \quad (2.76h)$$

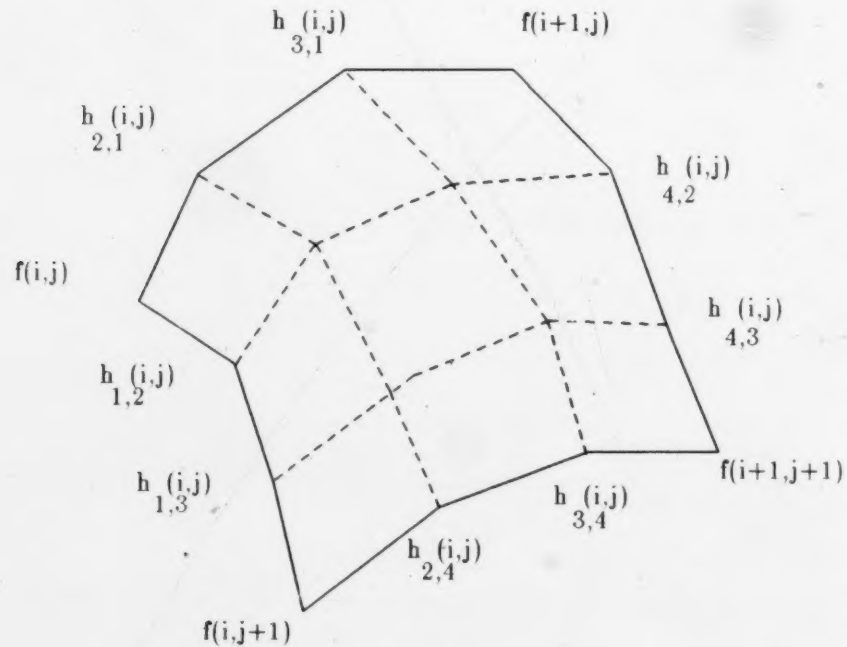


Figure 2.8 Guiding Points.

$$\delta_{i,j} = 3(3(h_{2,2}(i,j) - h_{1,2}(i,j)) - 3(h_{2,1}(i,j) - f(i,j))) \quad (2.76i)$$

$$\delta_{i,j+1} = 3(3(h_{2,4}(i,j) - f(i,j+1)) - 3(h_{2,3}(i,j) - h_{1,3}(i,j))) \quad (2.76j)$$

$$\delta_{i+1,j} = 3(3(h_{4,2}(i,j) - h_{3,2}(i,j)) - 3(f(i+1,j) - h_{3,1}(i,j))) \quad (2.76k)$$

$$\delta_{i+1,j+1} = 3(3(f(i+1,j+1) - h_{3,4}(i,j)) - 3(h_{4,3}(i,j) - h_{3,3}(i,j))) \quad (2.76l)$$

with the guarding points defined as in Fig. 2.6. It should be evident that the $h_{k,l}$'s are actually functions of i and j too.

Using equations (2.76) and (2.71)

$$B_{i,j}(s,y) = XM_h M_{hb} Q_{B_{i,j}} M_{hb}^T Y^T \quad (2.77)$$

where M_{hb} is defined in (2.48) as

$$M_{hb} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -3 & 3 & 0 & 0 \\ 0 & 0 & -3 & 3 \end{bmatrix}$$

$Q_{B_{i,j}}$ is the 4 by 4 matrix of gray values and guiding points:

$$Q_{B_{i,j}} = \begin{bmatrix} f(i,j) & h_{1,2}(i,j) & h_{1,3}(i,j) & f(i,j+1) \\ h_{2,1}(i,j) & h_{2,2}(i,j) & h_{2,3}(i,j) & h_{2,4}(i,j) \\ h_{3,1}(i,j) & h_{3,2}(i,j) & h_{3,3}(i,j) & h_{4,3}(i,j) \\ f(i+1,j) & h_{4,2}(i,j) & h_{4,3}(i,j) & f(i+1,j+1) \end{bmatrix} \quad (2.78)$$

and $0 \leq x, y \leq 1$, $i = 0, 1, \dots, m-2$ and $j = 0, 1, \dots, m-2$. In total it is sufficient if the values in Table 2.2 are given. The bicubic spline interpolation defines the free parameters $d_{i,j}$, $\partial_{i,j}$ and $\delta_{i,j}$, $i = 0, 1, \dots, m-1$ and $j = 0, 1, \dots, n-1$ as derivatives with (2.73), (2.74), and (2.75) in such a way that the interpolant g is twice differentiable everywhere. It becomes apparent that as with the one dimensional spline functions, certain values of the derivatives on the boundary of the rectangle must be given in addition to the gray values.

Table 2.2 The Required Initial Conditions:

x-direction							
			0	1	...	$m-1$	
		$\delta_{0,0}$	$d_{0,0}$	$d_{1,0}$	...	$d_{m-1,0}$	$\delta_{m-1,0}$
y	0	$\partial_{0,0}$	$f(0,0)$	$f(1,0)$	...	$f(m-1,0)$	$\partial_{m-1,0}$
d i r	1	$\partial_{0,1}$	$f(0,1)$	$f(1,1)$	...	$f(m-1,1)$	$\partial_{m-1,1}$
	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
	$n-1$	$\partial_{0,n-1}$	$f(0,n-1)$	$f(1,n-1)$	...	$f(m-1,n-1)$	$\partial_{m-1,n-1}$
		$\delta_{0,n-1}$	$d_{0,n-1}$	$d_{1,n-1}$	...	$d_{m-1,n-1}$	$\delta_{m-1,n-1}$

where

$$d_{i,j} = \frac{\partial q}{\partial y}|_{(i,j)} \quad (2.73)$$

$$\partial_{i,j} = \frac{\partial q}{\partial x}|_{(i,j)} \quad (2.74)$$

$$\delta_{i,j} = \frac{\partial^2 q}{\partial y \partial x}|_{(i,j)} \quad (2.75)$$

for $i = 0, 1, \dots, m-1$, $j = 0, 1, \dots, n-1$.

Corresponding to the system of equations (2.53), the partial derivatives $d_{i,j}$, $\partial_{i,j}$ and $\delta_{i,j}$ are found as follows: by fixing a parameter of bicubic spline interpolant, $g(x, y)$ and varying the other parameter a spline is obtained. Let y take on the value of some j , $0 \leq j \leq n-1$, then the cubic spline must be twice differentiable with respect to x . Therefore, as in (2.50) and (2.51), given the boundary conditions $\partial_{0,j}$ and $\partial_{m-1,j}$,

$$\partial_{i-1,j} + 4\partial_{i,j} + \partial_{i+1,j} = 3(f(i+1,j) - f(i-1,j)), \quad (2.79)$$

for $i = 0, 1, \dots, m-2$ is obtained.

Analogously, when the parameter x takes on some value i , $0 \leq i \leq m-1$, the second-order continuity of the spline results in

$$d_{i,j-1} + 4d_{i,j} + d_{i,j+1} = 3(f(i,j+1) - f(i,j-1)), \quad (2.80)$$

where $j = 0, 1, \dots, m-2$ and the boundary conditions $d_{i,0}$ and $d_{i,n-1}$ are given.

The $\delta_{i,j}$ are determined in a similar manner: the values $\delta_{i,j}$, $i = 0, 1, 2, \dots, m-1$, $j = 0$ and $j = n-1$, are calculated by fitting two sets of splines through $d_{i,j}$ for $i = 0, 1, 2, \dots, m-1$, $j = 0$ and $j = n-1$ using the boundary values $\delta_{0,0}$, $\delta_{m-1,0}$, $\delta_{0,n-1}$ and $\delta_{m-1,n-1}$. These newly calculated $\delta_{i,j}$ values are then used to find the remaining $\delta_{i,j}$ for $i = 0, 1, 2, \dots, m-1$, $j = 1, 2, \dots, n-2$ by interpolating $\partial_{i,0}$, $\partial_{i,1}$, $\dots$, $\partial_{i,n-1}$ for each $i = 0, 1, \dots, m-1$. This approach assumes that

$$\frac{\partial^2 q}{\partial x \partial y} = \frac{\partial^2 q}{\partial y \partial x}, \quad (2.81)$$

which is valid if the surface $g(x,y)$ is known to be continuous and x and y are continuous variables. Since the gray values in an image are usually not continuous and the variables are definitely not continuous, the above scheme for finding the net of $\delta_{i,j}$'s cannot be used. Instead, the values for $\delta_{i,0}$ and $\delta_{i,n-1}$ for $i = 0, 1, \dots, m-1$ are assumed to be given and they are used to calculate the remaining $\delta_{i,j}$ for $i = 0, 1, 2, \dots, m-1$, $j = 1, 2, \dots, n-2$ by interpolating $\partial_{i,0}$, $\partial_{i,1}$, $\dots$, $\partial_{i,n-1}$ for each $i = 0, 1, \dots, m-1$. This implies that for $i = 0, 1, \dots, m-1$,

$$\delta_{i,j-1} - 4\delta_{i,j} + \delta_{i,j+1} = 3(\partial_{i,j+1} - \partial_{i,j-1}) \quad (2.82)$$

with $j = 1, 2, \dots, n-2$.

Thus, the preceding procedure is equivalent to solving for:

(i) $\partial_{i,j}$ for $i = 1, 2, \dots, m-2$, $j = 0, 1, \dots, n-1$ after determining the boundary conditions $\partial_{0,j}$ and $\partial_{m-1,j}$ using (2.79).

(ii) $d_{i,j}$ based on (2.80) and the given values of $d_{i,0}$ and $d_{i,n-1}$ where $i = 0, 1, \dots, m-1$ and $j = 1, 2, \dots, n-2$.

(iii) $\delta_{i,j}$ using (2.82) and the given values $\delta_{i,0}$, $\delta_{i,n-1}$, $\delta_{0,n-1}$ for $i = 0, 1, \dots, m-1$ and $j = 1, 2, \dots, n-2$.

Like (2.53), each of the above operations will result in a set of systems with tridiagonal, symmetric diagonally dominant matrix so that solvability and solution are guaranteed. There are a total of $2m + n$ systems since i varies integrally from 0 to $m-1$ in (2.80) and (2.82) while j takes on n values in (2.79). In addition to the required values as shown in Table 2.2, an additional $2m-4$ $\delta_{i,j}$'s must be known before the above systems can be solved, i.e., $\delta_{i,0}$ and $\delta_{i,n-1}$ for $i = 1, 2, \dots, m-2$. The computed $d_{i,j}$, $\partial_{i,j}$, and $\delta_{i,j}$ values are then used in (2.71) to compute $g(x, y)$, for $0 \leq x \leq m-1$ and $0 \leq y \leq n-1$.

In summary, to find the bicubic interpolants for a surface over a finite rectangular region, the values of the parameters $d_{i,j}$, $\partial_{i,j}$ and $\delta_{i,j}$ must be known in addition to $f(i, j)$, for $0 \leq i < m$ and $0 \leq j < n$.

The Hermite form requires that these three classes of parameters be defined as in (2.73), (2.74) and (2.75), whereas the Bezier bicubic introduces control points $h_{k,l}(i, j)$ to determine the parameters $d_{i,j}$, $\partial_{i,j}$, and $\delta_{i,j}$ using (2.76). The bicubic spline which offers second order continuity defines $d_{i,j}$, $\partial_{i,j}$ and $\delta_{i,j}$ in (2.79), (2.80) and (2.82). Except for the Bezier bicubic, the others require the estimation of $\frac{\partial q}{\partial x}$, $\frac{\partial q}{\partial y}$ and $\frac{\partial^2 q}{\partial y \partial x}$. Apparently the use of the Bezier form will then overcome the need for approximations. On examining (2.76), however, it is clear that the guiding

points play the role of the derivatives; the points enable the definition of tangents. Consequently, the user has either to interactively enter these points or an estimation scheme has to be implemented. In this thesis, the Bezier is not studied further. The derivatives are approximated using the Δ 's in (2.58) - (2.60) for the Hermite and spline interpolants.

2.4. Fourier Transforms of the Basis Functions of the Interpolation

One convenient way to examine the validity of the schemes introduced in the previous section is to determine the Fourier transforms and compare them with the rectangle function $R(s)$. The analysis for two dimensional interpolation extends naturally from the following one dimensional results. For the ensuing discussion, let $g(x)$ be the interpolated function based on equally spaced samples of the continuous function $f(x)$. In particular, the equally spaced sampling points are

$$x_0, x_1, \dots, x_n.$$

2.4.1. Linear

Rewriting (2.35) for linear interpolation,

$$g(x) = \sum_{p=0}^{n-1} \left\{ f(x_p) \frac{(x_{p+1}-x)}{h} + f(x_{p+1}) \frac{(x-x_p)}{h} \right\} \Upsilon(x, p) \quad (2.83)$$

where $\Upsilon(x, p) = U(x-x_p)U(x_{p+1}-x)$. This product of two step functions ensures that the correct interval for x is considered. If $x_{p+1}-x_p = h$ for $p = 0, 1, \dots, n-1$, then

$$g(x) = \sum_{p=0}^{n-1} f(x_p) T\left(\frac{x-x_p}{h}\right) \quad (2.84)$$

where $T(x)$ is the Chateau function. Taking the Fourier transform of both sides of (2.84) results in

$$F\{g(x)\} = \sum_{p=0}^n f(x_p) F\left\{T\left(\frac{x-x_p}{h}\right)\right\}. \quad (2.85)$$

Using the shift invariant property of the Chateau function with the shifting theorem and the fact that

$$F\left\{T\left(\frac{x}{h}\right)\right\} = h \operatorname{sinc}^2(hs), \quad (2.86)$$

the corresponding frequency domain relation for (2.84) is

$$G(s) = h \operatorname{sinc}^2(hs) \sum_{p=0}^n f(x_p) e^{-i2\pi \frac{x_p}{h}s}. \quad (2.87)$$

Evidently, the filter allows leakage of high frequencies since $\operatorname{sinc}^2(hs)$ is sinusoidal with infinite duration, i.e., the product in (2.87) contains portions of high frequencies attributed by the replicas in $F(s)$. These frequencies cause aliasing. Recall that the ideal interpolation filter is the $R(s)$ function.

2.4.2. Piecewise Cubic Interpolation

Using the definition of the interpolating cubic polynomial $p_i(x-x_i)$ for the interval $[x_i, x_{i+1}]$ as defined in (2.39), $g(x)$ can be rewritten as

$$\begin{aligned} g(x) &= \sum_{i=0}^{n-1} p_i(x-x_i) \Upsilon(x, i) \\ &= \sum_{i=0}^{n-1} [a_{1i}(x-x_i)^3 + a_{2i}(x-x_i)^2 + a_{3i}(x-x_i) + a_{4i}] \Upsilon(x, i). \end{aligned} \quad (2.88)$$

Using the Shifting Theorem, the Fourier transform of $g(x)$ is therefore

$$G(s) = \sum_{j=0}^{n-1} \{e^{-i2\pi x_j s} [a_{1j} \int_0^h x^3 e^{-i2\pi s x} dx \quad (2.89)$$

$$\begin{aligned} &+ a_{2j} \int_0^h x^2 e^{-i2\pi s x} dx + a_{3j} \int_0^h x e^{-i2\pi s x} dx \\ &+ a_{4j} \int_0^h e^{-i2\pi s x} dx]\}. \end{aligned} \quad (2.90)$$

The integrals in the above equation can be defined using a recurrence relation [22] by use of the gamma function,

$$\Gamma(k) = \int_0^h x^k e^{-i2\pi sz} dz. \quad (2.91)$$

Values for $\Gamma(k)$ are given by

$$\Gamma(0) = \frac{1 - e^{-i2\pi sh}}{i2\pi s}, \quad (2.92a)$$

$$\Gamma(k) = \frac{1}{i2\pi s} [k\Gamma(k-1) - h^k e^{-i2\pi sh}], \quad k > 0. \quad (2.92b)$$

The coefficients a_m ($m=0,1,2,3$) in (2.89) and (2.90), that are evaluated according to the respective interpolation schemes, determine the amount of filtering of $F(s)$ in the frequency domain. Although theoretically (2.89) reveals the behavior of $G(s)$ in the frequency domain, it is deceptively difficult to display. Nevertheless, the aliasing effect can be investigated by examining the behavior of $G(s)$ for large magnitudes of s .

On expanding the recurrence relations (2.92) for $k = 1, 2, 3, \dagger$

$$\Gamma(1) = \frac{-2i\pi s + e^{-2i\pi sh} + 2he^{-2i\pi sh} i\pi s}{4\pi^2 s^2}, \quad (2.93)$$

$$\Gamma(2) = \frac{1}{4i\pi^3 s^3} [-2i\pi s + e^{-2i\pi sh} + 2he^{-2i\pi sh} i\pi s - 2h^2 e^{-2i\pi sh} \pi^2 s^2], \quad (2.94)$$

$$\Gamma(3) = \frac{1}{8\pi^4 s^4} [6i\pi s + 3e^{-i2\pi sh} + 6he^{-2i\pi sh} i\pi s - 6h^2 e^{-2i\pi sh} \pi^2 s^2 - 4h^3 e^{-2i\pi sh} i\pi^3 s^3]. \quad (2.95)$$

The difficulties in gaining an insight into $G(s)$ can be attributed to the representations of the cubic interpolants. It is possible to express the cubic Hermite interpolation and splines as linear combinations of some set of basis functions in $2n+2$ dimensional vector space with respect to $x_0, x_1, \dots, x_n$, i.e.,

$$g(x) = \sum_{p=0}^n [f(x_p) b_p(x) + f'(x_p) \tilde{b}_p(x)] \quad (2.96)$$

where $b_p(x)$ and $\tilde{b}_p(x)$ are the basis functions [36,39]. Following the analysis for

[†] These equations were derived using MAPLE.

linear interpolation using (2.85), it is then possible to express $G(s)$ by finding the Fourier transforms of the basis functions.

2.4.3. Two Dimensional Interpolation

The bilinear interpolation can be written as

$$g(x, y) = \sum_{i=0}^n \sum_{j=0}^n f(x_i, y_j) T\left(\frac{x-x_i}{h}, \frac{y-y_j}{h}\right) \quad (2.97)$$

where $f(x, y)$ is sampled at (i, j) , $i = 0, 1, 2, \dots, n$ and $j = 0, 1, 2, \dots, n$ [35].

The separability of $T(x, y)$ into $T(x)$ and $T(y)$, and the knowledge of the Fourier transform of the Chateau function allows the Fourier transform of $g(x, y)$ to be expressed as

$$G(s, t) = h^2 \text{sinc}^2(hs) \text{sinc}^2(ht) \sum_{p=0}^n \sum_{q=0}^n f(x_p, y_q) e^{-i2\pi(\frac{x_p}{h} + \frac{y_q}{h})} \quad (2.98)$$

$G(s, t)$ for the bicubic Hermite interpolation and splines can be similarly determined once the Fourier transforms of the basis functions are known. That is,

$$G(s, t) = \sum_{p=0}^n \sum_{q=0}^n f(x_p, y_q) \mathbf{F}\{\tilde{b}_{p,q}(x, y)\} \quad (2.99)$$

where $\tilde{b}_{p,q}(x, y)$ are the basis functions. $G(s, t)$ can then be compared with $\mathbf{F}\{f(x, y)\}R(s, t)$.

Chapter 3

Smoothing Techniques

In this chapter, smoothing and image enhancement techniques for reducing aliasing in an image are examined. These methods can be roughly divided into two categories: frequency domain methods and spatial domain methods. In frequency domain methods, the image is transformed into the frequency domain (e.g., using a transform such as Fourier) and further processing like low pass filtering is performed in that domain before the image is converted back to the spatial representation. On the other hand, spatial domain methods are based on direct manipulation of the pixel values themselves.

In the following discussions it is assumed that the image to be smoothed is a k times expansion of a given image by replicating each pixel in the original image to form a $k \times k$ block. Adjacent blocks with different gray levels give rise to irregular edges. This appearance may be improved if the pixels near these edges are given transition gray values that incorporate the nature of the neighboring blocks. Smoothing is one way of adjusting the gray values of these pixels. Spatial domain smoothing techniques that can be used to smooth out the aliasing are described in the following section. Most spatial smoothing processes involve an averaging operation of intensity values of neighboring pixels to reduce the contrast between adjacent pixels in the image. This is evident from the diminished "sharpness" in features like edges, lines and curves. Some of the smoothing methods are combined with image enhancement techniques and adapted to smooth across the blocks. Sharpening methods are discussed in Chapter 4.

3.1. Spatial Domain Smoothing and Noise-Cleaning Methods

Smoothing is usually applied to reduce the presence of noise in an image. The most straightforward method is local averaging. For an $n \times n$ image $f(x,y)$, each pixel at (x,y) is given a new gray value $g(x,y)$ which is the average of the original gray levels in some neighborhood of (x,y) , i.e.,

$$g(x,y) = \frac{1}{|S|} \sum_{(i,j) \in S} f(i,j) \text{ for } x,y = 0,1,\dots,n-1 \quad (3.1)$$

and S is a predefined neighborhood of (x,y) with $|S|$ equal to the number of pixels in it [14].

For points in a uniform region, averaging preserves the local average and reduces the variability. Thus, isolated noise points can be suppressed; however, this operation tends to blur the image. If applied indiscriminantly, sharp details will be degraded at the same time that false edges between blocks are weakened. For example, lines or edges are "smeared" by averaging across them.

There is a wide variety of variations to this simple approach. Each method adopts a decision criterion for averaging at each point based on the variability of the point from its surrounding neighbors.

3.1.1. Weighted Averaging

Local averaging over a fixed neighborhood as described above is only effective for a homogeneous image, i.e., an image with few local variations. To keep the blur to a tolerable level, small neighborhoods can be used.

Alternatively, weighted averagings based on the pixels in the neighborhood may also be employed. Two common weighted averaging schemes are listed below [30]. Assuming that P is the point at (x,y)

$$g_1(x, y) = \frac{1}{10} \left[\sum_{(i,j) \in N_8(P)} f(i, j) + 2f(x, y) \right] \quad (3.2)$$

$$g_2(x, y) = \frac{1}{16} \left[\sum_{(i,j) \in N_4(P)} 2f(i, j) + \sum_{(i,j) \in ND(P)} f(i, j) + 4f(x, y) \right] \quad (3.3)$$

where $ND(P)$ is $N_8(P) - N_4(P)$.

1	1	1
1	2	1
1	1	1

Figure 3.1 Mask with weighting defined in (3.2)

1	2	1
2	4	2
1	2	1

Figure 3.2 Mask with weighting defined in (3.3)

Effectively, (3.2) defines a mask of 3×3 pixels as shown in Fig. 3.1 and (3.3) represents the mask in Fig. 3.2. Averaging can be implemented by centering the respective mask over each pixel in the image and multiplying each weight in the mask with the underlying pixel. Weights lying outside the mask region are assumed to be zeros so that the sum of the products is the weighted sum. The divisor in each average can be considered as a normalization factor for the mask; a brightness bias, caused by using the non-fractional weights, will not be introduced in the processed image. These masks take into consideration the intensity values of the eight neighbors of P so as to blend it into its surrounding pixels. The first mask assumes an equal effect from each of the eight neighbors, but the second mask incorporates the difference in the distance between P and its vertical or horizontal neighbors and

the distance between P and its diagonal neighbors.

Smoothing with such a mask can therefore, be described by discrete convolutional filtering, i.e.,

$$g_2(x, y) = \sum_{i=1}^x \sum_{j=1}^y f(i, j) H(x-i+1, y-j+1) \quad (3.4)$$

where the input image is m by n pixels so that $1 \leq x \leq m$ and $1 \leq y \leq n$. The impulse response arrays H are the respective masks in Fig. 3.1 and 3.2 with the weights normalized.

More generally, a larger mask could be employed to include a bigger region for averaging. The disadvantage with this is the increase in the amount of computation. Moreover, an image with a large number of regions and features, can be adversely blurred such using larger masks.

3.1.2. Local Averaging with Thresholding

The demanding computational needs for a smoothing scheme where each pixel is processed can be reduced by limiting the smoothing operation to those pixels that are in neighborhoods with sufficient gray level variations.

In local averaging using thresholding, the gray values are changed only if they are substantially different from those in their neighborhood. In other words, if a pixel is in a fairly uniform region, smoothing is not performed. The processed image function $g(x, y)$ can thus be expressed in terms of the original image function $f(x, y)$ as follows:

$$g(x, y) = \begin{cases} \frac{1}{|S|} \sum_{(i, j) \in S} f(i, j) & \text{if } |f(x, y) - \frac{1}{|S|} \sum_{(i, j) \in S} f(i, j)| > T \\ f(x, y) & \text{otherwise.} \end{cases} \quad (3.5)$$

There are 2 parameters that can be adjusted to suit the application: S , the

neighborhood and T , the threshold value. Depending on the nature of the image, the size of S will affect the degree of blur in the new image. Although the average

$$\frac{1}{|S|} \sum_{(i,j) \in S} f(i,j) \quad (3.6)$$

is indicative of the general gray levels in the neighborhood S , it does not discriminate between points in the vicinity of edges or lines and noise points. This can be partially overcome if instead of comparing $f(x,y)$ with the average gray level in S , $f(x,y)$ compared with $f(i,j)$ for each $(i,j) \in S$ and using the count on the number of major deviations as the deciding criterion. In fact, some of the following methods employ this idea. In this case, T is the difference between $f(x,y)$ and the standard deviation of the pixels in S [32].

3.1.3. Variable Neighborhood Averaging

The size of S in (3.5) can be used to control the degree of smoothing. A variable neighborhood size can be used for each point in the operation. One approach is to consider growing neighborhoods around a point until the sum of the gray values in the final neighborhood reaches a threshold [32]. The neighborhood will be large if the gray level sum is low and vice-versa. The average gray level is then inversely proportional to S . The degree of smoothing is therefore dependent on the threshold value, and the uniformity and gray values in the vicinity of a point. An alternative to this non-uniform smoothing scheme is to expand neighborhoods until the number of gray levels that are substantially different from the point reaches a threshold. For this approach, the neighborhood will be large if the point is in a uniform region and small if the region has a high gray level variability. This method assumes a high degree of confidence that a point belongs to a surrounding region if the gray levels around it are fairly uniform. Thus smoothing

based on a larger neighborhood will blend the point into the region and is least likely to blur an edge or line. Conversely, if the local area is highly non-uniform, it is unlikely for the point to be in the interior of a region and smoothing is kept to a minimum to avoid effects of blurring.

3.1.4. Selective Neighbor Averaging

Instead of using the size of the neighborhood of a pixel $f(x,y)$ to effect the amount of smoothing, the k pixels with gray levels nearest to $f(x,y)$ in a fixed size neighborhood can also be employed [7]. Usually the size of the neighborhood is chosen to be 3×3 and $k = 2, 4, 6$ are commonly used. In particular when $k = 2$, edges and lines passing through a 3×3 neighborhood are preserved best since the two pixels adjacent to $f(x,y)$ have the two closest gray values.

3.1.5. Maximum Homogeneity Smoothing

One selective approach is to examine a set of fixed neighborhoods around a point and the average gray level of the region with the least gray level variability is given to that point. The motivation is the likelihood that the point is in that region. Tsuji et. al. [6,39] have proposed a set of five 3×3 square neighborhoods around each point (x,y) for consideration. As shown in Fig. 3.3, the five neighborhoods are centered at x , x_1 , x_3 , x_5 and x_7 . Nagao et. al. [6,25], however, has observed that any 3×3 region will be merged into the surrounding areas after several iterations of the smoothing procedure. In addition, independent regions with false gray levels can arise. To see why these distortions occur, consider the 3×3 region, R centered at x as in Fig. 3.3 and suppose the set of 5 3×3 regions centered at x is denoted as $SR(x)$. Each of the points x_1 , x_3 , x_5 and x_7 in R has at least one 3×3 defined neighborhood from the corresponding set $SR(x_i)$ for

$i = 1, 3, 5, 7$, that is completely included in R . In contrast, none of the 3×3 neighborhoods of $SR(x_i)$, for $i = 0, 2, 4, 6$, are totally contained in R . That is, each of the neighborhoods in $SR(x_i)$, for $i = 0, 2, 4, 6$ contain part of R and the surrounding region, so that the points x_0, x_2, x_4 and x_6 will be given a different gray level than that of R . After several iterations, the gray levels at these points will approach that of the surrounding region, merging R into it.

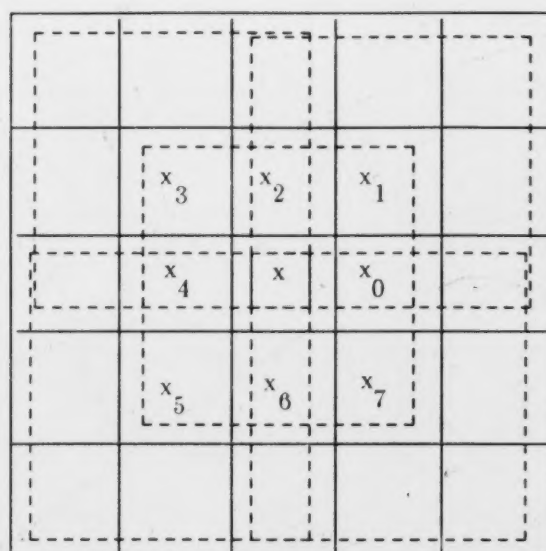


Figure 3.3 5 3×3 neighborhoods centered at (x, y)

3.1.6. Maximum Homogeneity Smoothing with Masks

To locate the most homogeneous region around a point P , Nagao [25] has proposed using pentagonal and hexagonal masks. In Fig. 3.4, a pentagonal X and a hexagonal T neighborhood is shown. The average gray level and variance of each neighborhood defined by the respective mask are computed and P is given the average gray value of the region with the lowest variance.

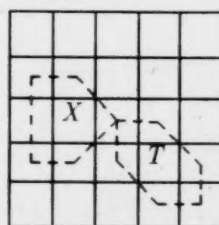


Figure 3.4 Pentagonal and Hexagonal Masks

3.1.7. Mode and Median Filtering

For a neighborhood in which the gray levels are fairly uniform, the mode in the neighborhood can also be used [32]. Yet another smoothing technique is to use the local median [30] instead of the local mean. This operation does not blur edges because no averaging is done and the median is not affected if an edge passes through a fraction of the neighborhood. That is, the median is not one of those pixels belonging to such a passing edge. On the other hand, thin lines, corners and isolated points are eliminated. This is illustrated in Fig. 3.5. The figure on the right is the filtered version of the figure on the left.

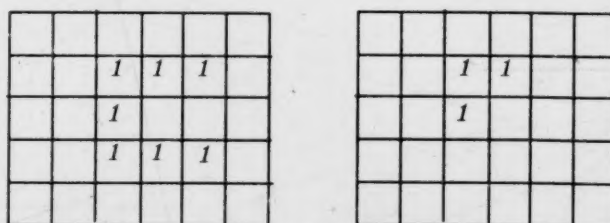


Figure 3.5 Median Filtering

3.1.8. Edge or Line Smoothing

Another selective averaging method is to attempt to use the underlying image structure to guide the local averaging process. One could apply a set of edge detection operations to detect the presence of lines and edges and then average pixels where these features are absent or the pixels are on the same side of the edge. This avoids smoothing across detected features. An alternative to choosing these points, one may select the neighborhood consisting of those pixels along an edge or line. One simple method is to use a set of masks to detect edges and lines in the vicinity of a point P . The gray level is the average of the two adjacent pixels to P along the direction of an edge or line if either feature is determined to pass near P or, otherwise the average of a fixed neighborhood is assigned to it. There is a variety of edge detectors. In general, they resemble the derivatives of the intensity functions. This is because the number of edges, lines or noise points present in an image is proportional to the rate of change of the intensity function. The two common detectors are the gradient and the Laplacian of the image functions. Again, there are several forms and each can be characterized [24,31].

Both the effectiveness and efficiency of such smoothing schemes depends on the size of the neighborhood that is used. If only the eight immediate neighbors of a point P is used, then the new gray value of P is estimated using at most nine points (fewer if an edge or curve passes through P). A better estimate could be obtained if a larger neighborhood is used since more of the pixels from the same region as P are employed in the evaluations, but then the complexity of detecting edges and curves becomes greater, since there are many ways in which an edge or curve can pass through a large neighborhood. A compromising approach is to use a small neighborhood, but to iterate the smoothing operation. In other words, an estimate

of the new gray level of each pixel P is obtained, and the process is repeated using these new gray values in place of the original gray levels, which amounts to weighted averaging.

3.1.9. Half Neighborhood Methods

Alternatively to considering the gradient and Laplacian at each pixel, P , the possibility of a passing edge through a neighborhood of P could be considered. If there is sufficient evidence of such an edge, the point P is averaged with the pixels on the same side of the edge. This can be accomplished by considering the eight neighbors of P . These eight neighbors are divided into sets of two groups of consecutive pixels. Let $N_1(z_i)$ be the group with 5 points $z_i, z_{(i+1) \bmod 8}, \dots, z_{(i+4) \bmod 8}$, and $N_2(z_{(i+5) \bmod 8})$ with 3 points, $z_{(i+5) \bmod 8}, z_{(i+6) \bmod 8}$ and $z_{(i+7) \bmod 8}$, for $i = 0, 1, \dots, 7$. The region containing P is the group $N_1(z_i)$ of the eight possible choices of i , with the nearest average gray value to that of P . The presence of an edge is then decided by the magnitude of $|\bar{N}_1(z_i) - \bar{N}_2(z_{(i+5) \bmod 8})|$ where the value for $\bar{N}_j(z_i)$ is the average gray value for $N_j(z_i)$ for $j = 1, 2$ and $0 \leq i \leq 7$. For some preset threshold T , if $|\bar{N}_1(z_i) - \bar{N}_2(z_{(i+5) \bmod 8})| > T$, it is then assumed that an edge passes through the neighborhood of P and P is most likely to be in a region together with $N_1(z_i)$. Thus, P is smoothed with the points of $N_1(z_i)$. On the other hand, if $|\bar{N}_1(z_i) - \bar{N}_2(z_{(i+5) \bmod 8})| \leq T$, P is inside a bigger region and the average gray level of $N_1(z_i) \cup N_2(z_{(i+5) \bmod 8}) \cup P$ for some $i = 0, 1, \dots, 7$ (or $N_8(P) \cup P$) is used [34].

A different criterion for deciding the presence of a region edge is to choose the largest $|\bar{N}_1(z_i) - \bar{N}_2(z_{(i+5) \bmod 8})|$ and using $|\bar{N}_1(z_i) - \bar{N}_2(z_{(i+5) \bmod 8})| > T$ to determine the presence of an edge. In effect, the strongest possible edge is chosen and if it presents adequate evidence, P is given the average of the gray levels of the

pixels in $N_1(x_i) \cup P$. Otherwise, P is given the average of the gray levels of pixels in $N_1(x_i) \cup N_2(x_{(i+5) \bmod 8}) \cup P$. Instead of selecting consecutive neighbors around a point P , one may select k (e.g. $k = 5$) neighbors which have the closest gray level to that of P . Intuitively, P is in the same region as the k neighbors [34].

In most of the above methods, weights can be assigned to each neighbor of P according to the degree of likelihood it is in the same region as with P . For example, when P is given the $\bar{N}_1(x_i) \cup P$ the implicit weight of $\frac{1}{6}$ is given to each pixel in $N_1(x_i) \cup P$ and 0 to those in $N_2(x_{(i+5) \bmod 8})$. It has been found that methods which incorporate a weighting scheme do not produce better results than schemes which choose the best neighborhood and ignore the others [32].

3.1.10. Histogram Smoothing

The histogram of an image measures the relative frequencies of all possible gray levels used in the quantization. The probabilities of the gray levels of the neighbors of P can be used based on the histogram as selection criteria. If an edge passes near a point P , then the neighbors with similar probabilities as P would more likely to be on the same side of P , so that P can be averaged with those values. Still another alternative is to select neighbors with the k highest probabilities, for $k = 5$ or 7. In other words, the points inside a region are used.

The histogram based on the gray levels of those pixels in the neighborhood of a pixel P can also be used to determine a subset of pixels whose gray levels are closest to that of P . In particular, if the neighborhood histogram is at least bimodal, the neighbors whose gray levels belong to the same peak of the histogram could be used as that of P itself.

3.1.11. Directional Smoothing

Following along the lines of using the gradient or Laplacian to indicate the amount of edge information in the vicinity of a pixel at (x, y) , Rosenfield and Kak [6, 32] have proposed using four other measures. In the first method, d_1 and d_2 are defined as in (3.7) and (3.8). If $\max\{d_1, d_2\}$ exceeds some threshold T_1 , then the average of the two pixels in the direction $\tan^{-1} \frac{d_1}{d_2}$ evaluated to the nearest 45° , is used to replace $f(x, y)$. Otherwise, the average of all the pixels in the 3×3 neighborhood of (x, y) is used. It should be noted that $\tan^{-1} \frac{d_1}{d_2}$ is perpendicular to the direction of the greatest change in gray levels since it approximates the gradient.

Alternatively, d_3 and d_4 as defined in (3.9) and (3.10) can be used in addition to d_1 and d_2 . If $\max\{d_1, d_2, d_3, d_4\} - \min\{d_1, d_2, d_3, d_4\}$ exceeds the preset value T_2 , the two pixels in the direction of minimum absolute gray level difference is used.

$$d_1 = \frac{1}{3} |(f(x-1, y+1) + f(x, y+1) + f(x+1, y+1)) - (f(x-1, y-1) + f(x, y-1) + f(x+1, y-1))| \quad (3.7)$$

$$d_2 = \frac{1}{3} |(f(x-1, y-1) + f(x-1, y) + f(x-1, y+1)) - (f(x+1, y-1) + f(x+1, y) + f(x+1, y+1))| \quad (3.8)$$

$$d_3 = \frac{1}{3} |(f(x-1, y) + f(x-1, y+1) + f(x, y+1)) - (f(x, y-1) + f(x+1, y-1) + f(x+1, y))| \quad (3.9)$$

$$d_4 = \frac{1}{3} |(f(x, y-1) + f(x-1, y-1) + f(x-1, y)) - (f(x+1, y) + f(x+1, y+1) + f(x, y+1))| \quad (3.10)$$

3.1.12. Gradient Smoothing

The gradient of a pixel P indicates the greatest magnitude and direction of intensity change at P . The gradient is small for P if it is in a homogeneous region and similarly, the neighbors of P will have small gradient values. Thus, when P is in such a region, replacing the gray value of P by the average of those k pixels with lower gradient magnitudes than P will smooth P with other pixels in the region. On the other hand, if P is near an edge, pixels that are used in the averaging will be from the interior of the regions adjacent to the edge. This will result in blurring of the edge. In order to limit this blurring, the direction of the vector (gradient) should be taken into consideration.

Another approach is to use those k pixels in the neighborhood of P with the closest gradient values. In this case, if P is near an edge, those pixels near the edge are averaged with P . Clearly, the problem of edge blur in the above method is imminent. The same remedy of taking the direction into consideration should be employed.

3.1.13. Laplacian Smoothing

The Laplacian is a rotation-invariant second order difference which has zero response to linear ramps, but responds to the top and bottom of the ramp where there is a change in the rate of change in gray level. Using the same approach as in gradient smoothing, one can smooth a pixel with the k neighboring pixels with similar Laplacian as P . In this case if P is in a region, the response is small (or zero) and therefore it will be averaged with pixels in the same homogeneous region. Similarly, if P is near an edge the Laplacian will be large and using those pixels with similar Laplacian value, those pixels near the same side of edge are used to

smooth P. The advantage of using the Laplacian over the gradient is that the polarity of the Laplacian of a pixel can be used to indicate its location relative to an edge whereas, the direction has to be evaluated in addition to the magnitude of the gradient.

3.1.14. Directionally Sensitive Weighted Averaging

Lev et. al. have proposed two types of weighted averaging that are directionally sensitive [6, 23]. Two types of weighting matrices were used: line sensitive and contrast sensitive weights. For the former type, twelve 3×3 matrices, A_i , $1 \leq i \leq 12$, are defined as in Fig. 3.6 to enumerate all possible cases of a possible edge or line.

$$\begin{aligned}
 A_1 &= \begin{bmatrix} a & a & a \\ b & b & b \\ b & b & b \end{bmatrix}, \quad A_2 = \begin{bmatrix} a & a & b \\ a & b & b \\ b & b & b \end{bmatrix}, \quad A_3 = \begin{bmatrix} a & b & b \\ a & b & b \\ a & b & b \end{bmatrix}, \\
 A_4 &= \begin{bmatrix} b & b & b \\ a & b & b \\ a & a & b \end{bmatrix}, \quad A_5 = \begin{bmatrix} b & b & b \\ b & b & b \\ a & a & a \end{bmatrix}, \quad A_6 = \begin{bmatrix} b & b & b \\ b & b & a \\ b & a & a \end{bmatrix}, \\
 A_7 &= \begin{bmatrix} b & b & a \\ b & b & a \\ b & b & a \end{bmatrix}, \quad A_8 = \begin{bmatrix} b & a & a \\ b & b & a \\ b & b & b \end{bmatrix}, \quad A_9 = \begin{bmatrix} b & b & b \\ a & a & a \\ b & b & b \end{bmatrix}, \\
 A_{10} &= \begin{bmatrix} b & b & a \\ b & a & b \\ a & b & b \end{bmatrix}, \quad A_{11} = \begin{bmatrix} b & a & b \\ b & a & b \\ b & a & b \end{bmatrix}, \quad A_{12} = \begin{bmatrix} a & b & b \\ b & a & b \\ b & b & a \end{bmatrix}.
 \end{aligned}$$

Figure 3.6 The twelve possible edge/line configuration.

$$\begin{aligned}
D_1 &= \frac{1}{6} \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, & D_2 &= \frac{1}{6} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, & D_3 &= \frac{1}{6} \begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}, \\
D_4 &= \frac{1}{6} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}, & D_5 &= \frac{1}{6} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}, & D_6 &= \frac{1}{6} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \\
D_7 &= \frac{1}{6} \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}, & D_8 &= \frac{1}{6} \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}, & D_9 &= \frac{1}{3} \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \\
D_{10} &= \frac{1}{3} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, & D_{11} &= \frac{1}{3} \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}, & D_{12} &= \frac{1}{3} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.
\end{aligned}$$

Figure 3.7 The corresponding weight matrices.

Corresponding to these twelve matrices A_i , for $1 \leq i \leq 12$, are twelve weight matrices D_i , shown in Fig. 3.7. For each pixel at (x, y) in the image a 3×3 neighborhood A , centered at (x, y) , is considered. In the presence of noise, A will rarely match any of the A_i 's and even if it does, it may be due to some noise artifact. A weight matrix to be used for the averaging at a pixel is defined as

$$D = \sum_{i=1}^{12} w_i D_i \quad (3.11)$$

where the w_i 's represent a measure of resemblance between A and A_i , for $1 \leq i \leq 12$.

The elements of this weight matrix are thus dynamically altered for each pixel depending on the intensity values of the pixels in its neighborhood. If $f(x, y)$ is the intensity at (x, y) , the following are defined to find each w_i : $S(A)$ is the sum of the gray levels of the nine pixels in A , $S_j(A)$ is the sum of the gray levels of those three pixels in A located at the "a" position in A_j , and $\Theta_j(A_i)$ is $3S_j(A_i) - S(A_i)$. Then, one can observe that the following relations hold:

$$\Theta_j(A_i) = 6(a - b) \quad (3.12a)$$

$$\Theta_j(A_i) = 3(a-b) \text{ if } A_i \text{ and } A_j \text{ are both edges and have adjacent slopes (3.12b)}$$

$$\text{i.e., } 1 \leq i, j \leq 8 \text{ and } |(i-j)| = 1. \quad (3.12c)$$

$$\Theta_j(A_i) \leq 0 \text{ otherwise (assuming } a > b),$$

for $1 \leq i, j \leq 12$.

Under the assumption that $a > b$, $\Theta_j(A)$ is therefore large if A resemble A_j .

The w_i 's are then defined as

$$w_i = \frac{\Theta_i^+(A)}{\sum \Theta_j^+(A)} \quad (3.13)$$

where

$$\Theta_k^+(A) = \begin{cases} \Theta_k(A) & \text{if } \Theta_k(A) \geq 0 \\ 0 & \text{if } \Theta_k(A) < 0, \end{cases} \quad (3.14)$$

where $k = i$ or j . One should note that the weights w_i 's are not sensitive to gray levels of the pixels in A since the factor $(a-b)$ cancels from the numerator and denominator when $A = A_i$. The w_i 's depend only on the orientation of the ideal edges and lines that match A .

In the second scheme, a 3×3 adaptive weight matrix is defined for a pixel P by considering its 3×3 neighborhood and setting the weights in the matrix to be small if there is a strong presence of an edge or line. The small value prevents the gray value of the corresponding pixel to have adverse effect on the averaging. Four separate weighting matrices which are sensitive to edges in the four principal directions are defined. The elements in each matrix are inversely proportional to the edge strength at each corresponding point in the 3×3 neighborhood.

The final weighting matrix is then defined from the above four separate matrices by taking the product of the corresponding elements.

Shown below are D_0, D_1, D_2, D_3 and the final matrix D_f .

$$D_0 = \begin{bmatrix} \alpha & \alpha & \alpha \\ 1 & 1 & 1 \\ \beta & \beta & \beta \end{bmatrix}, \quad D_1 = \begin{bmatrix} \gamma & \gamma & 1 \\ \gamma & 1 & \delta \\ 1 & \delta & \delta \end{bmatrix}$$

$$D_2 = \begin{bmatrix} \epsilon & 1 & \xi \\ \epsilon & 1 & \xi \\ \epsilon & 1 & \xi \end{bmatrix}, \quad D_3 = \begin{bmatrix} 1 & \eta & \eta \\ \theta & 1 & \eta \\ \theta & \theta & 1 \end{bmatrix},$$

$$D_f = \begin{bmatrix} \alpha\gamma\epsilon & \alpha\gamma\eta & \alpha\xi\eta \\ \gamma\epsilon\theta & 1 & \delta\xi\eta \\ \beta\epsilon\theta & \beta\delta\theta & \beta\delta\xi \end{bmatrix},$$

Figure 3.8 Weight Matrices.

where $\alpha = e^{-\frac{|(P_1+P_2+P_3)-(P_0+P+P_4)|}{\sigma}}$, $\beta = e^{-\frac{|(P_6+P_6+P_7)-(P_0+P+P_4)|}{\sigma}}$,
 $\gamma = e^{-\frac{|(P_2+P_3+P_4)-(P_1+P+P_5)|}{\sigma}}$, $\delta = e^{-\frac{|(P_6+P_7+P_0)-(P_1+P+P_5)|}{\sigma}}$,
 $\epsilon = e^{-\frac{|(P_3+P_4+P_5)-(P_2+P+P_6)|}{\sigma}}$, $\xi = e^{-\frac{|(P_1+P_0+P_7)-(P_2+P+P_6)|}{\sigma}}$,
 $\eta = e^{-\frac{|(P_2+P_1+P_0)-(P_3+P+P_7)|}{\sigma}}$, $\theta = e^{-\frac{|(P_4+P_6+P_6)-(P_3+P+P_7)|}{\sigma}}$, σ is the standard deviation of the intensities in the 3×3 neighborhood and
 $N8(P) = \{P_0, P_1, \dots, P_7\}$.

3.2. Adaptive Smoothing and Noise-Cleaning Techniques

Most techniques have been shown to achieve reasonably good results in conditions of weak noise. These techniques are used to smooth a k -expanded image to produce a k -times smoothed expansion of an image. The principal difficulty in this approach is for large k , smoothing is often restricted to pixels from two adjacent blocks. For small values of k most averaging techniques use pixels from several surrounding blocks, which allows better blending effects. As k increases and with averaging neighborhoods remain unchanged, the number of blocks from which the pixels are used in the smoothing decreases. Therefore, for larger values of k , smoothing operations are not effective. This problem can be alleviated by using larger averaging neighborhoods. For example, larger masks could be employed for

weighted averaging using masks. An immediate consideration is the redefinition of the masks and the increase in computation time.

The need to incorporate the intensities of blocks in the vicinity during smoothing can be partially accomplished by fixing the order in which pixels are averaged and the intensities values used. For each block, pixels can be smoothed in layers: those pixels lying at the edges of the blocks are averaged first, following by those adjacent to them and inside the same block, and so on. In addition, when a pixel is given a new intensity value, this value is subsequently used by all other averaging that requires it. By doing so, gray values of surrounding blocks are cascaded into the interior of the blocks, allowing more gray transitions.

An extension to this method is to partition each block of $k \times k$ pixels into sub-blocks. For example, each k -block can be divided into $9 \frac{k}{3} \times \frac{k}{3}$ sub-blocks. Smoothing is then performed using the sub-blocks as the atomic entities rather than pixels so that pixels in a sub-block are given the same gray value. This scheme can be iterated on the sub-blocks for further smoothing. The disadvantage of dividing a k -block into 9 smaller ones is that the frequently employed image dimensions are not divisible by three. On the other hand, by employing this partitioning method, a 360° view of the surrounding blocks is available.

Another approach to enhance the effectiveness of smoothing schemes for large expansions is to expand and smooth the image in small steps. For example, a k -expansion can be effected by performing a series of smaller expansions like $\frac{k}{i}$ expansions, assuming that k is divisible by $\frac{k}{i}$.

Chapter 4

IMAGE SHARPENING

Sharpening techniques are useful to improve the fidelity of an image by highlighting edges in the image. In Chapter 3, it was seen that most smoothing schemes are accomplished by some form of averaging and the resulting image is generally blurred. Therefore, it is natural to expect to use techniques which require differentiation operations to sharpen an image. This chapter considers the most basic techniques for improving edge quality in the spatial domain.

4.1. Gradient

One of the most popular differential operators is the gradient. The gradient of the intensity function f of the continuous variables x, y is defined as the vector

$$\text{grad } f(x, y) = \frac{\partial f}{\partial x} \hat{i} + \frac{\partial f}{\partial y} \hat{j} \quad (4.1)$$

where $\hat{i}$ and $\hat{j}$ are the unit vectors in the x and y directions, respectively. This vector is in the direction of the maximum rate of change of f and its magnitude is given by: $[(\frac{\partial f}{\partial x})^2 + (\frac{\partial f}{\partial y})^2]^{1/2}$. In practice, the direction of the gradient is usually ignored and only the magnitude is used. This terminology will be employed in the following discussion.

There are several discrete approximations of the gradient available [14, 32] :

1. Robert's approximation:

$$\text{grad } f(x, y) = ([f(x, y) - f(x+1, y+1)]^2 + [f(x+1, y) - f(x, y+1)]^2)^{1/2}, \quad (4.2)$$

or with further approximation

$$\text{grad } f(x, y) = |f(x, y) - f(x+1, y+1)| + |f(x+1, y) - f(x, y+1)| \quad (4.3)$$

2. another common approximation is

$$\text{grad } f(x,y) = [[f(x,y)-f(x+1,y)]^2 + [f(x,y)-f(x,y+1)]^2]^{\frac{1}{2}}, \quad (4.4)$$

or using absolute values

$$\text{grad } f(x,y) = |f(x,y)-f(x+1,y)| + |f(x,y)-f(x,y+1)|. \quad (4.5)$$

It can be noted that the above approximations are based on intensity differences in adjacent pixels. In the vicinity of edges, the gradient is large and by the same token, the gradient is small in homogeneous regions and the background. Thus, edge presence is highlighted by these operators.

Intuitively, this information can be added back in the original image, to increase the contrast between edges and regions.

Again, there are many ways this can be implemented. If $g_1(x,y)$ is the intensity at (x,y) , define

$$g_1(x,y) = \begin{cases} f(x,y) + \text{grad } f(x,y) & \text{if } f(x,y) + \text{grad } f(x,y) \leq \text{max gray level} \\ \text{max gray level} & \text{otherwise} \end{cases} \quad (4.6)$$

or,

$$g_2(x,y) = \begin{cases} g_1(x,y) & \text{if } \text{grad } f(x,y) \geq T_2 \\ f(x,y) & \text{otherwise} \end{cases} \quad (4.7)$$

The first definition is the simple minded approach with clipping to avoid excessively large intensity values to occur. The second method allows emphasis on the more significant edges and ignores the others.

4.2. Laplacian

The Laplacian is the linear difference operator [14]

$$\nabla^2 f(x, y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}. \quad (4.8)$$

Rosenfeld and Kak [32] have modeled image blur as a diffusion process and shown the application and relevance of the Laplacian for picture sharpening.

There are three basic discrete versions of this operator:

$$\nabla_1^2 f(x, y) = 5f(x, y) - [f(x-1, y) + f(x+1, y) + f(x, y-1) + f(x, y+1)], \quad (4.9)$$

$$\nabla_2^2 f(x, y) = 9f(x, y) - \sum_{(i, j) \in N8((x, y))} f(i, j) \quad (4.10)$$

$$\nabla_3^2 f(x, y) = 5f(x, y) + \sum_{(i, j) \in ND((x, y))} f(i, j) - 2 \sum_{(i, j) \in N4((x, y))} f(i, j) \quad (4.11)$$

where $ND((x, y))$ is defined to be $N8((x, y)) - N4((x, y))$.

These approximations relate the Laplacian of an intensity function f proportionally to the difference between f and an average version of f . For a more detailed discussion see [32].

These operators can also be defined as impulse response arrays with

$\nabla_1^2 f(x, y)$, $\nabla_2^2 f(x, y)$ and $\nabla_3^2 f(x, y)$, respectively correspond to the masks

$$M_1 = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix},$$

$$M_2 = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix},$$

$$M_3 = \begin{bmatrix} 1 & -2 & 1 \\ -2 & 5 & -2 \\ 1 & -2 & 1 \end{bmatrix}.$$

This correspondence enables sharpening to be performed in the frequency domain by using the convolution theorem for the Fourier transform: the output image is the inverse transformation of the product of the Fourier transforms of the input image and the mask.

Chapter 5

RESULTS AND ANALYSIS

From the spatial domain methods presented in Chapter 3, it is realized that the techniques are at best ad hoc. This conclusion is not restricted to this area of image processing. There are three major reasons for the lack of universality and elegance in the current solutions. First, there is not a single model by which to describe image enhancement; the actual models used are dependent on the nature of the processes by which the image was obtained. Even if the actual model is available, computational considerations usually require various approximations. Lastly, there is no universally agreed criteria by which the quality of the processed image can be judged. The two common approaches are based either on numerical measures between the processed and the original image or subjective evaluation of the resemblance between the processed and original images. Yet these two approaches more often than not result in conflict. These difficulties stem mainly from the adaptive psychophysical processes of human vision, which is not totally understood nor modeled.

In the following sections, the results of the implemented interpolation and noise-cleaning schemes are illustrated. Their computational requirements are also examined. These methods were implemented on a VAX 11/780 driving an International Imaging System model 75. Three images of different structures are used. To investigate the performance of these methods on images with many large regions, an image of the moon is used. The effect of these methods on lineal features are tested using an image with text. An image consisting of a figure is also used to test the validity of these methods. These three images will be referred to as

MOON, TEXT and FIGURE, respectively. It shall be assumed in the following discussion that an n by n image is expanded k times.

5.1. Pixel Replication.

Plate 5.1 contains the 4 times (linear) expansion of 64x64 original. The advantage of this method is that no calculation is needed to generate the $k^2 n^2$ pixels.

5.2. Interpolations.

The following interpolation methods are implemented:

1. Bilinear.
2. Bilinear with Median Filtering - the original image is always 2-expanded and the edge pixels in each 2x2 block are calculated using bilinear interpolation and the centre pixel is the median of these new eight edge pixels.
3. Piecewise Bicubic Hermite.
4. Natural Spline.
5. Complete Spline.

The derivatives in methods 3 to 5 can be specified to be one of $\Delta_1 f$, $\Delta_2 f$ and $\Delta_3 f$ as defined by (2.58) - (2.60).

5.2.1. Bilinear Interpolation.

The expansion of a n^2 image by k in each dimension requires $O(k^2 n^2)$ multiplications and additions since there are $k^2 n^2 - n^2$ points to be interpolated. Instead of using (2.35), the technique of forward differences [41] can be used to reduce the number of multiplications.

Plate 5.2 shows the 4-times bilinear expansions of the MOON, TEXT and FIGURE. As one might expect, image blur is considerable when k is large. One measure E_I proposed by Pratt [30] is defined as:

$$E_I = \frac{(E_T - E_R)}{E_T} \quad (5.1)$$

and

$$E_T = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \hat{F}(s,t) G(s,t) ds dt, \quad (5.2)$$

$$E_R = \int_{-c_1}^{c_1} \int_{-c_2}^{c_2} F(s,t) G(s,t) ds dt, \quad (5.3)$$

where $\hat{F}(s,t)$ and $F(s,t)$ are the spectra of the sampled and original images, respectively, and c_1, c_2 are the cutoff frequencies $\frac{1}{\tau_x}, \frac{1}{\tau_y}$ with τ_x and τ_y being the sampling intervals in the x and y directions. $G(s,t)$ represents the reconstruction filter or transfer function and E_I is the amount of energy located beyond the Nyquist band limits [30].

The additional cost in the bilinear with filtering approach is attributed by the sorting required to locate the median of the 3×3 neighborhoods. Using quicksort, the increase is $O(n^2)$ for the entire image.

5.2.2. Piecewise Hermite Interpolation

The number of multiplications for evaluating the gray level of each pixel using (2.79) and (2.81) is $2.4^3 + 2.4^2 + 2.M$ where M is the number of multiplications required to find the first derivative for the Hermite geometry vector. Therefore, if computation is done according to (2.76) and (2.78) the total number of multiplications involved will be $(2.4^3 + 2.4^2 + 2.M)(k^2n^2 - n^2)$. Using (2.59) to evaluate the derivative will cost 1 subtraction whereas (2.60) will need 1 multiplication and 1 addition. (2.61) requires 2 multiplications and 2 additions per evaluation. Although the number of multiplications is $O(k^2n^2)$, the hidden constant is large.

Forward (finite) differences and Horner's method can be used to cut down the number of operations. Plates 5.4 to 5.6 illustrate the piecewise Hermite interpolation. The presence of block artifacts, which are more prominent than in bilinear interpolation, can be attributed to the errors in estimating the derivatives.

On examining (2.79) and (2.80), any error in approximating the derivatives will be amplified. Suppose the errors in the estimation of $d_{i,j}$, $\delta_{i,j}$ and $\partial_{i,j}$ are $e_{i,j}^1$, $e_{i,j}^2$ and $e_{i,j}^3$, respectively. Following the definitions in (2.73) and (2.75),

$$d_{i,j} = \frac{\partial f}{\partial y}(x,y)|_{(i,j)} + e_{i,j}^1, \quad (5.4)$$

$$\partial_{i,j} = \frac{\partial^2 f}{\partial y \partial x}(x,y)|_{(i,j)} + e_{i,j}^2, \quad (5.5)$$

$$\delta_{i,j} = \frac{\partial f}{\partial x}(x,y)|_{(i,j)} + e_{i,j}^3, \quad (5.6)$$

so that

$$B_{i,j}(x,y) = XM_h(Q_{i,j} + \hat{E}_{i,j})M_h^T Y^T, \quad (5.7)$$

where $Q_{i,j}$ is the matrix with correct values (or values when the functions used are C^2) and $\hat{E}_{i,j}$ is the error matrix due to the estimations.

The error introduced is therefore $XM_h \hat{E}_{i,j} M_h^T Y^T$. To keep the notation simple, suppose $e_{r,s}^m = 0$ for all valid m, r, s except for $e_{i,j}^1 = e$. It then follows that

$$B_{i,j}(x,y) = XM_h Q_{i,j} M_h^T Y^T + (x^3 - 2x^2 + x)e(y^3 - 2y^2 + y). \quad (5.8)$$

Thus, e is magnified by $(x^3 - 2x^2 + x)(y^3 - 2y^2 + y)$ times.

5.2.3. Bicubic Spline Interpolation.

Straightforward calculation requires the solution of:

1. n systems of $n-2$ linear equations, each of which is defined by (2.79).
2. n systems, each of which is defined by (2.80).
3. 2 systems as defined by (2.57), for natural spline interpolation.
4. n systems as defined by (2.82).

The computations listed above are for finding the values of the elements of the matrix $Q_{i,j}$ in (2.69). The general Gaussian elimination takes $O(n^3)$ multiplications and additions to solve an $n \times n$ system [1]. The fact that the systems are tridiagonal, however, provides substantial savings. A tridiagonal system can be solved with $O(n)$ operations [5] so that the number of operations needed to solve all the above systems is $O(n^2)$. The calculation for the gray values will cost, in addition to the $O(n^2)$ operations, $(2.4^3 + 2.4^2)(k^2 n^2 - n^2)$ multiplications for evaluating (2.71). Therefore, the total number of operations is $O(k^2 n^2)$. Like the Hermite interpolation, it also suffers from the estimation of the derivatives.

5.3. Smoothing and Noise-Cleaning Techniques.

The smoothing and noise-cleaning techniques implemented are:

1. Neighborhood averaging using a square neighborhood with given size and threshold.
2. Averaging as in (1) except that a "circular" neighborhood is used.
3. Weighted averaging using masks as defined by (3.46) and (3.47).
4. As in (3) except that the pixels in each block are processed in layers.
5. Neighborhood averaging in which the neighborhood is expanded around each pixel until the sum of all pixels in the neighborhood has reached a given threshold. Only square neighborhoods are used.
6. Neighborhood averaging with growing neighborhood. Each neighborhood is grown until the number of intensity deviation from the processed pixel reaches a specified number. The threshold for determining a deviation is also given by the user.
7. Selective neighbor averaging as described in Section 3.1.4.
8. Tsuji's proposed method.
9. As in (9) but with "layered" order processing.
10. Nagao's proposed method.
11. As in (11) but with "layered" processing order.
12. Mode filtering.
13. Median filtering.
14. Half neighborhood schemes #1 and #2 as described in Section 3.1.9.
15. Histogram Smoothing in which neighbors in a 3×3 neighborhood with the k

highest frequencies are selected for averaging.

16. Directional smoothing schemes #1 and #2.
17. Gradient smoothing based on Robert's approximation.

Expansions are always carried out in two successive steps; thus, a 256×256 image is obtained by expanding a 64×64 original twice, by a magnitude of two each time. In addition to these noise-cleaning methods, five sharpening techniques described in Chapter 4 are also implemented. They are reiterated here for convenience:

1. Robert's approximation of the gradient using (4.2) and (4.7).
2. Approximation of the gradient using (4.3) and (4.7).
3. $\Delta_1 f$ as defined by (4.9) and (4.7).
4. $\Delta_2 f$ as defined by (4.10) and (4.7).
5. $\Delta_3 f$ as defined by (4.11) and (4.7).

5.3.1. Neighborhood Averaging.

There is little difference between square or circular neighborhood averaging. The blurring is more prominent for larger neighborhoods. For the $s \times s$ square neighborhood, the number of additions needed is $O(s^2 k^2 n^2)$ and the number of divisions is $k^2 n^2$. For circular neighborhood averaging with a neighborhood of radius r , the number of additions is $O(r^2 k^2 n^2)$.

5.3.2. Weighted Averaging.

The number of computations to use a 3×3 mask as in (3.2) or (3.3) is clearly $9k^2n^2$. Since the image to be smoothed is an expansion by pixel replication the smoothing based on (3.3) is not as effective as the one defined by (3.2). When the pixels at the edge of a block is smoothed, the proportionally larger weight of the centre pixel of the mask defined by (3.3) hinders the gray level of the surrounding pixels to play a larger role in the smoothing. Therefore, one should expect the resulting images due to the second scheme to contain more blocky artifacts than those processed by the first method. The images produced by the first method do not differ from those smoothed using circular or local averaging. Moreover, as mentioned in Chapter 3, when the expansion factor is large, the smoothing is less effective because pixels in the 3×3 mask are from fewer adjacent blocks.

Plates 5.13 - 5.16 show the smoothing in a straightforward manner and the "layered" methods. There is little visual improvement in these set of new images.

5.3.3. Variable Neighborhood Averaging.

It is difficult to count the number of operations since the size of the neighborhood is dependent on the threshold and the nature of the image. The amount of blur present in the resulting images obviously depends on the choice of the thresholds. Plate 5.17 illustrates a suitable threshold while the images in Plate 5.18 are excessively smoothed.

5.3.4. Selective Neighbor Averaging.

Again, the number of additions involved when m nearest gray level pixels as the processed pixel are used is mk^2n^2 and the number of divisions is k^2n^2 . Before these m pixels can be selected however, the pixels in each neighborhood must be sorted. The number of comparisons needed in the worst case when quicksort is used is 3^2c for some $c > 0$ [17], for 3×3 neighborhoods.

The results of using $m = 4$ for a 4-times expansion are generally blocky. This is, of course, due to the selection of pixels from the same blocks.

5.3.5. Tsuji's and Nagao's Schemes.

Tsuji's scheme requires $5(3^2 - 1)k^2n^2$ additions and k^2n^2 divisions while Nagao's method needs $8.7k^2n^2$ since hexagonal masks are used in addition to four pentagonal masks.

Both methods are based on finding a neighboring region with the lowest variance. Therefore, for a pixel located at the edge of its block, the gray level assigned to it is the average of a region situated in the identical block. Thus, the blocky artifacts due to pixel replications will not be removed by smoothing. This observation is evident in plate 5.21.

Since the gray levels of pixels near the edges of each block retain the original gray values, the "layered" processing order of the pixels will not reduced the aliasing.

5.4. Mode and Median Filtering.

In mode filtering, the frequencies of each gray level that occurs is noted and the gray level of the one with the highest frequency is assigned to the pixel which is being processed. This requires the registration of the frequencies which can be implemented using arrays and pointers. The task of locating the highest frequency requires a scan of all the frequency "buckets". No arithmetic operations is needed so that the number of operations is $O(s^2 k^2 n^2)$ comparisons when $s \times s$ neighborhoods are used.

On the other hand, the final scan for the highest frequency can be omitted if two additional variables are used. One of which can be used to always keep track of the highest frequency while the other is used as a pointer to the corresponding gray value.

Mode filtering tends to enlarge large regions and the results contain blocky artifacts. This is because the gray value of each block is given to the pixels at the edges of that block.

Apart from enlarging regions, median filtering removes isolated points and corners. Median filtering requires that the frequencies of the occurring gray levels be noted and the "buckets" sorted. Since each neighborhood is of fixed size, i.e., $s \times s$ the number of operations due to quicksort is $O(s^2)$ in the worst case. Huang et al. [19] has proposed a faster algorithm which will take $O(sk^2 n^2)$.

5.4.1. Half Neighborhood Methods.

The first scheme requires the determination of the half neighborhood $N_1(z_i)$ with $\bar{N}_1(z_i)$ nearest to the gray level of the pixel which is being processed. For each half neighborhood four additions and a division is needed. The nearest gray value can be found by determining the smallest absolute difference between the average gray values of the half neighborhoods and the pixel that is being processed. Hence, locating the proper half neighborhood will require sixteen additions/subtractions. The number of operations needed to process each pixel in the image is the same. Therefore, $O(k^2n^2)$ operations are needed. Again, depending on the value of the selected thresholds, the amount of smoothing can vary. Plate 5.27 illustrates a reasonable choice of the thresholds.

The second scheme locates the largest difference, $|\bar{N}_1(z_i) - \bar{N}_2(z_{(i+5) \bmod 8})|$ and checks to see if a given threshold is exceeded, in which case, averaging is performed using the pixels in $N_1(z_i)$. Clearly, this scheme will require a constant number of additional operations per pixel than the first scheme when the worst cases are considered. The operation count is therefore $O(k^2n^2)$, but with a larger constant than in the first scheme.

5.4.2. Histogram Smoothing.

In this method, as for mode and median filtering, the frequencies of gray values of pixels in a $s \times s$ neighborhood of each pixel is calculated. In selecting the gray levels, it is convenient to sort an array of absolute frequency differences between the processed pixel and the others in the neighborhood. The gray levels with the m frequencies nearest to that of the processed pixel can then be picked out using the results of the sorted array. For sorting s^2 elements using quicksort will require

$O(s^2)$ operations in the worst case. Therefore, the total operation count is $O((m + s^2)k^2n^2)$. The resulting images contain streaks of high intensities.

5.4.3. Directional Smoothing.

In the worst case, when the threshold is exceeded for each pixel, the operation count for either schemes is $O(k^2n^2)$. However, the second scheme will require a constant number of extra operations per pixels more than the first scheme. By choosing the proper thresholds, the amount of blur can be controlled.

5.4.4. Gradient Smoothing.

Like histogram smoothing, a way to find the gray levels of those pixels with gradients close to that of the pixel, P , which is being processed is to find the absolute difference in gradient between the pixels in the neighborhood and P . These differences are then sorted so that the pixels with their gradient closest to P can be easily selected. Since the size of each considered neighborhood is $s \times s$ and the cost of finding the gradient at each pixel is a constant, c , the number of operations is $O(s^2k^2n^2)$. Evidently, the constant c varies for different approximation schemes for the gradient.

5.5. Sharpening.

Shown in Plates 5.30 to 5.33 are the sharpened output of some of the expanded versions of MOON, TEXT and FIGURE. Following the reasoning as in the previous sections, the number of computations is clearly $O(k^2n^2)$. Since each method does not have an automatic thresholding capability, it is difficult to compare their merits. As seen in Plates 5.30 - 5.33, sharpening techniques with well chosen parameters can enhance a blurred expanded image.

5.6. Performance Measures.

The error measure used by Pratt [30] is based on the amount of energy introduced or removed from the original image by the interpolating filters. It is, however, difficult to find the spectrums of the noise-cleaning methods. Rather than using the measures described by Pratt, some basic measures are investigated as indicators for the performance of each of the implemented methods. The measures that are studied are all based on pixel differences of the original and processed images. These measures, therefore, do not take into consideration the context or content of the images. As mentioned before, there is no quantitative measure for the fidelity. The primary reason for examining these measures is to investigate the amount of pixel intensity deviation of the processed image from the original one. The measures that are considered can be classified into three types: the average intensity deviation per pixel, the average gray value deviation per gray level and the sum of the fractional deviations at each pixel. In addition, the logarithm or square of intensity differences are used to reflect the non-linear process of human perception. The measures evaluated for the expansion techniques are:

$$E_1 = \frac{\sum_i \sum_j |\ln(f(i,j)) - \ln(g(i,j))|}{\sum_i \sum_j 1}, \quad (5.9)$$

$$E_2 = \frac{\sum_i \sum_j \ln |f(i,j) - g(i,j)|}{\sum_i \sum_j 1}, \quad (5.10)$$

$$E_3 = \ln \left[\sum_i \sum_j \frac{|f(i,j) - g(i,j)|^2}{f(i,j)^2} \right], \quad (5.11)$$

$$E_4 = \frac{\sum_i \sum_j [\ln(f(i,j)) - \ln(g(i,j))]}{\sum_i \sum_j \ln(f(i,j))}, \quad (5.12)$$

$$E_5 = \frac{\sum_i \sum_j \ln[f(i,j) - g(i,j)]}{\sum_i \sum_j \ln(f(i,j))}, \quad (5.13)$$

$$E_6 = \ln \left[\sum_i \sum_j \frac{|\ln(f(i,j)) - \ln(g(i,j))|}{\ln(f(i,j))} \right], \quad (5.14)$$

$$E_7 = \ln \left[\sum_i \sum_j \frac{\ln |(f(i,j) - g(i,j))|}{\ln(f(i,j))} \right], \quad (5.15)$$

$$E_8 = \frac{\sum_i \sum_j |(f(i,j) - g(i,j))|}{\sum_i \sum_j f(i,j)} \times 100, \quad (5.16)$$

$$E_9 = \ln \left[\sum_i \sum_j \frac{|(f(i,j) - g(i,j))|}{f(i,j)} \right] \quad (5.17)$$

$$E_{10} = \left[\frac{\sum_i \sum_j [f(i,j) - g(i,j)]^2}{\sum_i \sum_j f^2(i,j)} \right] \times 100. \quad (5.18)$$

Thus, while Pratt's approach is a measure of the performance of the methods, the suggested schemes quantitatively evaluate the output images. Illustrated in Figs. 5.1 - 5.10 are the graphs of the measures for all the implemented expansion methods. The calculated value of E is based on a 4 times expansion of the 64×64 originals.

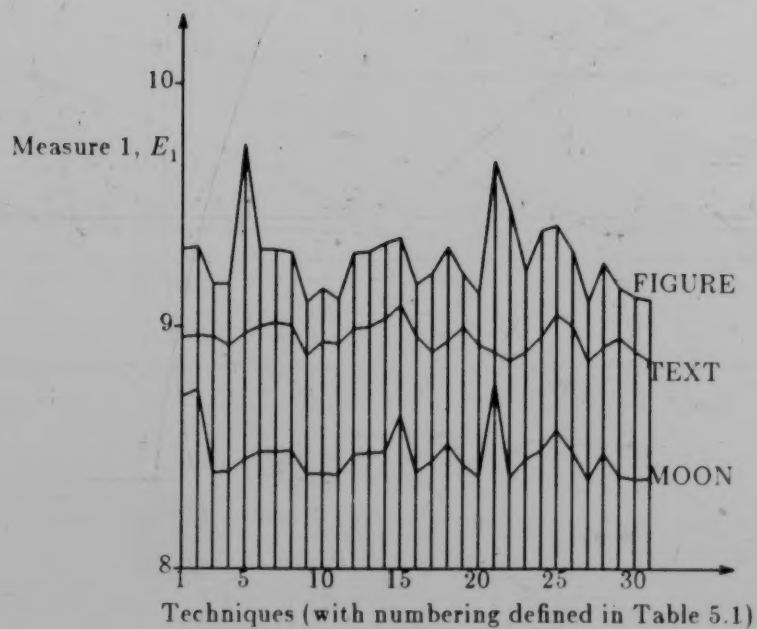


Figure 5.1 A Quantitative Comparison of the Expansion Techniques based on (5.9).

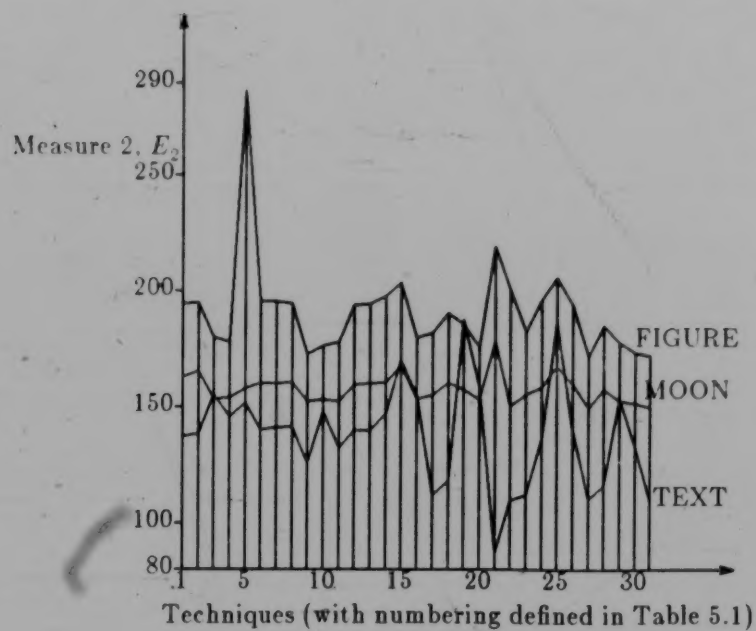


Figure 5.2 A Quantitative Comparison of the Expansion Techniques based on (5.10).

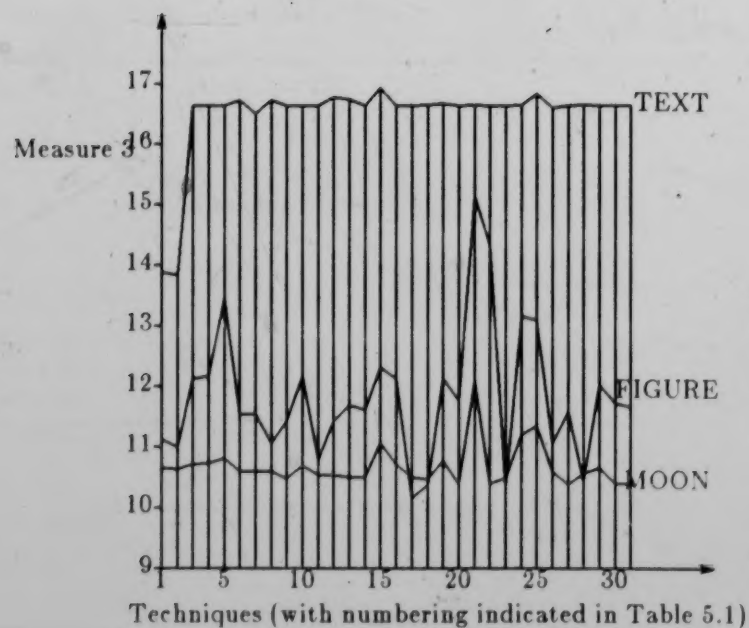


Figure 5.3 A Quantitative Comparison of the Expansion Techniques based on (5.11):

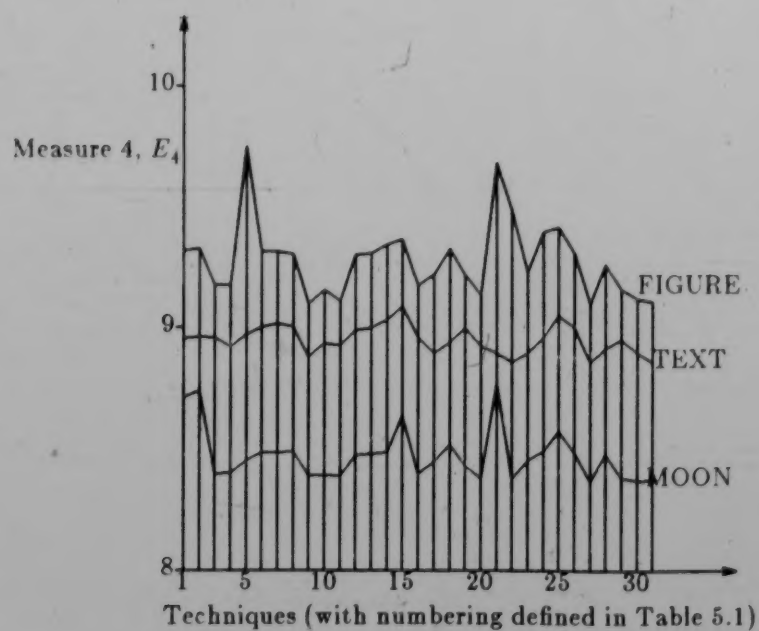


Figure 5.4 A Quantitative Comparison of the Expansion Techniques based on (5.12).

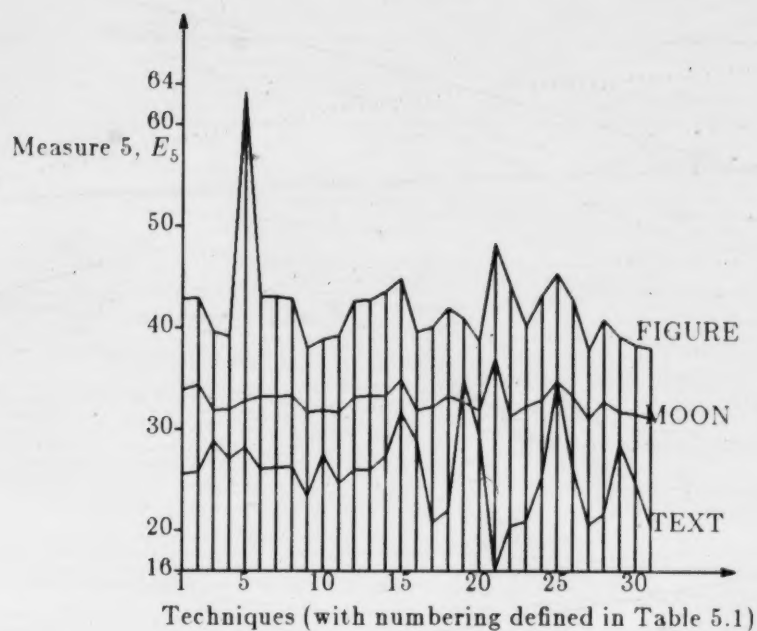


Figure 5.5 A Quantitative Comparison of the Expansion Techniques based on (5.13).

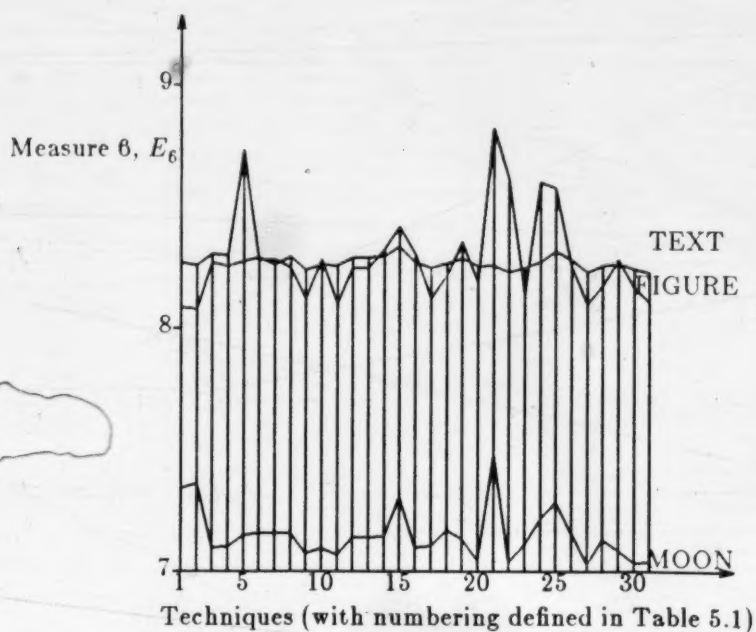


Figure 5.6 A Quantitative Comparison of the Expansion Techniques based on (5.14).

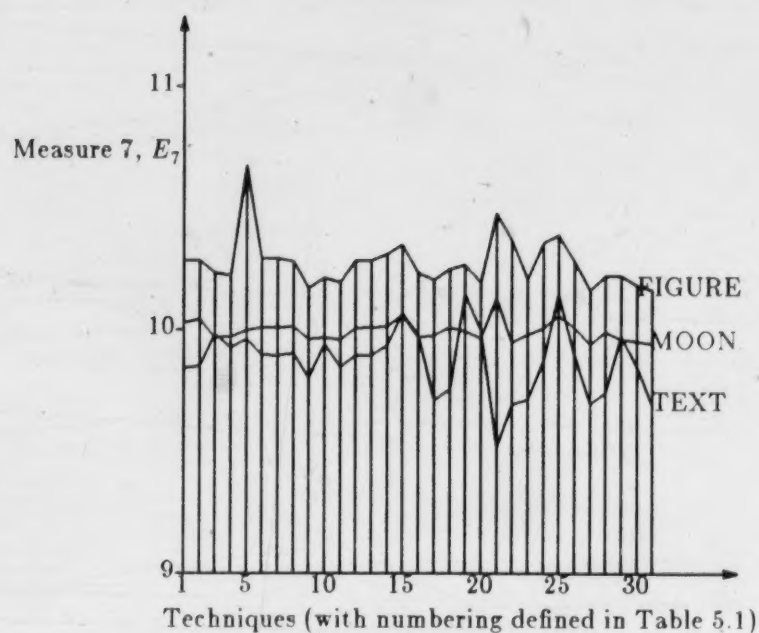


Figure 5.7 A Quantitative Comparison of the Expansion Techniques based on (5.15).

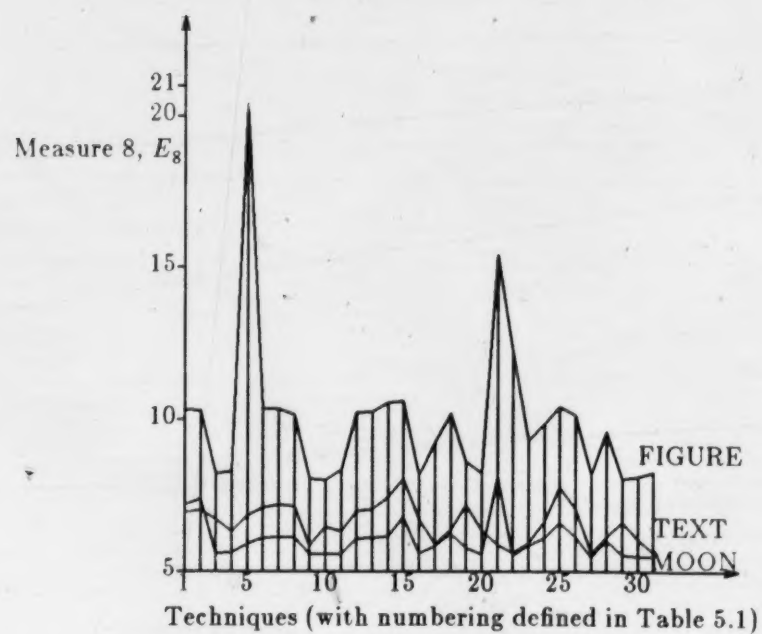


Figure 5.8 A Quantitative Comparison of the Expansion Techniques based on (5.16).

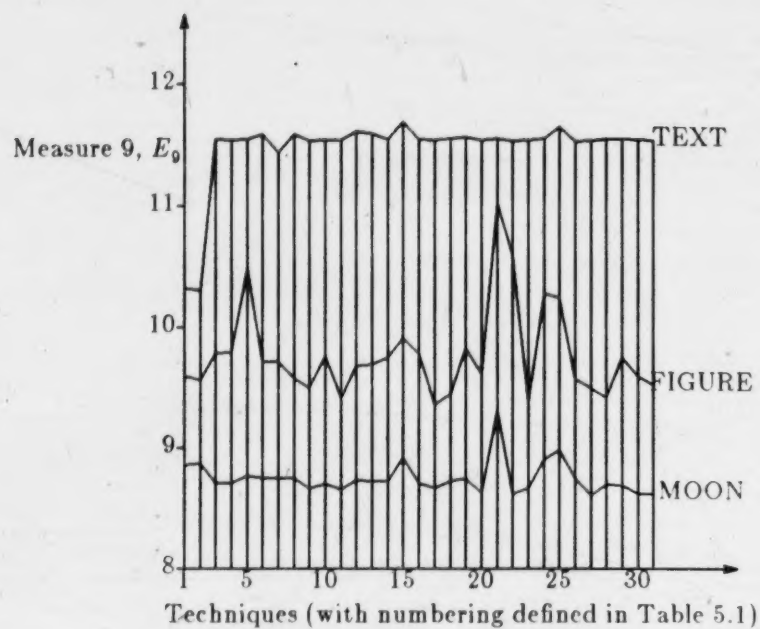


Figure 5.9 A Quantitative Comparison of the Expansion Techniques based on (5.17).

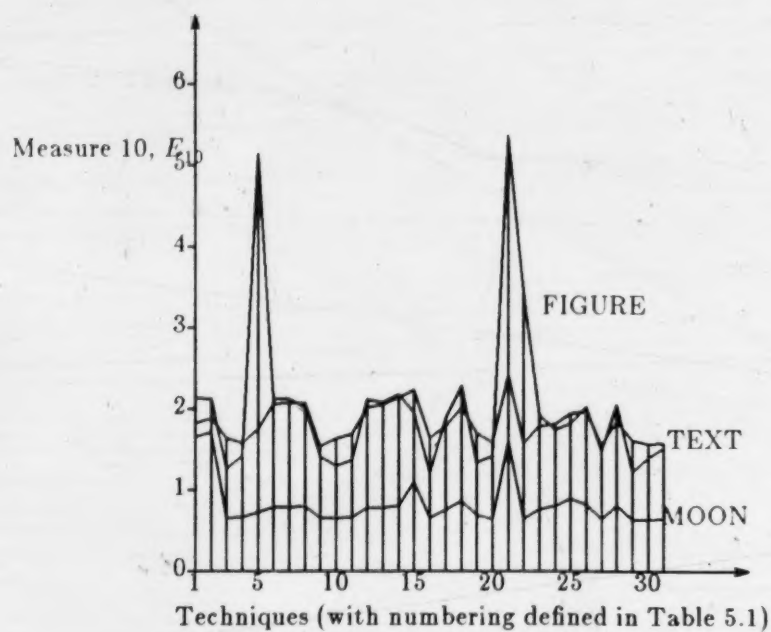


Figure 5.10 A Quantitative Comparison of the Expansion Techniques based on (5.18).

Since these measures are based on some kind of pixel difference between the original and processed images, a smaller resulting measure for an expansion technique should indicate a closer resemblance of the processed image to the original one. Thus, for a measure to be consistent with human visual evaluation (although it is debatable that human evaluation itself is consistent!) the values of most of the expansion methods should be smaller than the value for pixel replication. As seen from Figs. 5.1-5.10, none of the above measures is consistent with human evaluation. For example, the third measure E_3 indicates that the images processed using Tsuji's and Nagao's schemes are better than those processed by other methods. These simple measures suffers from two problems. Since the main concern is to measure pixel differences, the question of image alignment arises. Another concern is the possibility of overall intensity increase or decrease of the processed image. For example, if the weights in the mask used for weighted averaging is not normalized by dividing each weight by the sum of the weights in the mask, but by some other factor, an overall biased in the image intensity is resulted. Nevertheless, assuming that each measure is consistent for all expansion methods, it is possible to compare the performance of the expansion methods for different image contents using each of the measures. Since MOON has larger and more regions than TEXT, it is possible to infer from the relative positions of the TEXT and MOON lines in the graphs the effect of the expansion methods on them. For example, according to the third measure, the expansion methods work better on MOON than on TEXT.

Using the graphs, it is also possible to determine the deviation of the "layered" adaptations to the original techniques. From the graphs, the values due to the "layered" and the corresponding "unlayered" methods do not differ by much. This could imply that the performance of the adapted approach does not improve the

effect of the original method.

In general, the averaging methods produce reasonably good expansions. Their simplicity and specifiable parameters are suitable for interactive usage. Coupled with sharpening techniques, these methods produces images better than Tsuji's and Nagao's schemes, weighted averaging, mode and median filtering and half neighborhood averaging. The outputs of the other methods are comparable.

Images in which the significant features are lineal are not expanded properly. In particular, TEXT suffers from either insufficient smoothing which is suggested by the blocky nature of the image, or excessive smoothing in which lineal features are obscured.

The best interpolation technique is the bilinear method. The processing time is faster and the outputs are smoother than the bicubic methods. The natural and complete spline methods do not generate images with appreciable differences. Also, the outputs due to the bilinear and the averaging methods can suffer the same degree of blur. Most methods are seen to have $O(k^2n^2)$ time. This is deceptive since kn is bounded in practical applications; the resolution commonly used is seldom higher than 2^{14} . Thus, the missing constant in the asymptotic notation should be considered. For example, in the bicubic spline interpolations, the size of the constant is about 2^7 .

Table 5.1 Implemented Methods:

Expansion Technique	Num.	Expansion Technique	Num.
Bilinear	1	'Layered' Nagao	17
Modified Bilinear	2	'Layered' Tsuji	18
Circular Average (Rad. = 3, Thres = 0)	3	'Layered' Wt. #1	19
Half Neighborhood #1	4	'Layered' Wt. #2	20
Half Neighborhood #2	5	Median (3 x 3 neighborhood)	21
Complete Spline ($\Delta_1 f$)	6	Mode (3 x 3 neighborhood)	22
Complete Spline ($\Delta_2 f$)	7	Nagao	23
Complete Spline ($\Delta_3 f$)	8	Var. Neighd. #1 (Thres. = 500)	24
Directional #1 (Thres = 30)	9	Var. Neighd. #2 (3 x 3 neighd., # dev = 4)	25
Directional #2 (Thres = 50)	10	Natural Spline	26
Gradient (# pixels = 3)	11	Selective Averaging (# pixels = 4)	27
Hermite ($\Delta_1 f$)	12	Tsuji	28
Hermite ($\Delta_2 f$)	13	Weighted Averaging 1	29
Hermite ($\Delta_3 f$)	14	Weighted Averaging 2	30
Histogram (3 x 3 neighd., # pixels = 4)	15	Zoom	31
Local Averaging (3 x 3 neighd.)	16		



Plate 5.1 Replication: FIGURE, MOON and TEXT.



Plate 5.2 Bilinear Interpolation.



Plate 5.3 Modified Bilinear.



Plate 5.4 Piecewise Bicubic Hermite using $\Delta_1 f$.



Plate 5.5 Piecewise Bicubic Hermite using $\Delta_2 f$.



Plate 5.6 Piecewise Bicubic Hermite using $\Delta_3 f$.



Plate 5.7 Complete Spline using $\Delta_1 f$.



Plate 5.8 Complete Spline using $\Delta_2 f$.



Plate 5.9 Complete Spline using $\Delta_3 f$.



Plate 5.10 Natural Spline using $\Delta_3 f$.



Plate 5.11 Averaging using a 3×3 neighborhood.



Plate 5.12 Averaging using a circular neighborhood of radius 5.



Plate 5.13 Weighted Averaging using Mask #1.



Plate 5.14 Weighted Averaging using Mask #2.



Plate 5.15 'Layered' Weighted Averaging using Mask #1.



Plate 5.16 'Layered' Weighted Averaging using Mask #2.



Plate 5.17 Variable Neighborhood Scheme #1.



Plate 5.18 Variable Neighborhood Scheme #2.



Plate 5.19 Selective Neighborhood Averaging.



Plate 5.20 Histogram Smoothing.



Plate 5.21 Tsuji et.al. Noise-Cleaning Approach.



Plate 5.22 Nagao et.al. Noise-Cleaning Approach.



Plate 5.23 'Layered' Tsuji et.al. Method.



Plate 5.24 'Layered' Nagao et.al. Method.



Plate 5.25 Mode Filtering.



Plate 5.26 Median Filtering.



Plate 5.27 Half Neighborhood Noise-Cleaning Method #1.



Plate 5.28 Half Neighborhood Noise-Cleaning Method #2.



Plate 5.29 Gradient Smoothing.



Plate 5.30 TEXT: Smoothing followed by sharpening.



Plate 5.31 FACE: Smoothing followed by sharpening.

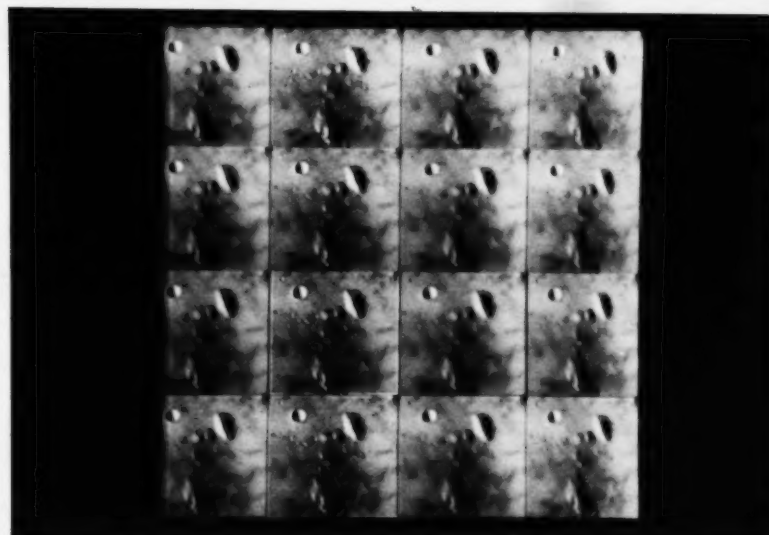


Plate 5.32 MOON: Smoothing followed by sharpening.



Plate 5.33 TEXT,FACE,MOON: Smoothing followed by sharpening.

Chapter 6

CONCLUSION

6.1. Overview

This research investigated the employment of interpolation and noise-cleaning methods for expanding images. In particular:

1. noise-cleaning and sharpening are surveyed in Chapter 3.
2. the theory of interpolation as a filtering process in the frequency domain is studied in Section 2.2.
3. the development of bicubic Hermite and spline interpolation are reviewed in Section 2.3.
4. a series of experiments is set up to show the feasibility of some of the smoothing and interpolation techniques for enlarging images from unrestricted environments.
5. the performance of the techniques are studied.
6. a software package which allow images to be expanded, smoothed and sharpened interactively was written.

6.2. Summary

Both interpolation and noise-cleaning classes of methods have been studied and used extensively in their own domains. Interpolation methods which are frequently applied in many areas of engineering, play a major role in image sampling, filtering and reconstruction. The advantage of using interpolation is the availability of mathematical tools for expressing the operations and manipulations.

This enables a concrete analysis of its merit. On the other hand, most smoothing methods are not as concrete in their rationale and frequently appeal to intuition. The apparent ad hoc nature is due to the lack of criteria for ranking images qualitatively. Statistical measures like mean square error have been used as qualitative indicators and are used as optimization criteria, but the applications are usually restricted. Without a total understanding or a proper model for human vision, it is difficult to find a set of criteria. With these setbacks, it is still interesting to investigate the applicability of the noise-cleaning methods in smoothing expanded images via pixel replication and their performance compared to interpolation methods.

In order to judge these methods on different sets of images a series of experiments were run. Based on the results the following general outlines can be stated:

1. Unless better approximation for derivatives in the discrete plane can be found, bilinear interpolation is better than bicubic interpolation in terms of speed and image fidelity. There are more sophisticated methods for approximating the derivatives. For example, the first derivative at x_0 can be determined by fitting a quadratic through the values at x_0 , x_1 and x_2 and estimating the first derivative at x_0 using this quadratic. An extreme technique for getting around the estimation of derivatives is to use (bi)cubic splines [18].
2. The simplest smoothing methods with properly selected thresholds provide better results than the other smoothing methods which are frequently more involved.
3. Noise-cleaning methods which choose a neighboring region that represents the most possible enclosure of the pixel which is being processed, do not provide good results. In particular, images from Tsuji's and Nagao's schemes are not

visually acceptable.

4. For large expansions, weighted averaging using masks do not smooth images adequately. In fact, methods that are based on a fixed (i.e., non-adaptive to the intensity levels of the image) neighborhood size do not smooth effectively for large expansions.
5. Mode and median filterings remove an undesirable amount of edge or line information.
6. The "layered" processing order of pixels do not provide significant improvement over the direct procedure. More effective smoothing is produced by breaking down a large expansion into smaller ones.
7. Except for mode and median filterings, all the implemented methods provide "smoother" renditions independent of the nature of the image.

In summary, this research has made an attempt to examine the validity and potential of noise-cleaning methods for image expansions. One of the prime objectives of this research is to design and implement a set of methods or rules that will categorize and expand images in each category into visually approvable ones. Fig. 6.1 illustrates the problem of image expansion and the techniques to approach it. Images can be categorized into binary and multi-level images. Simpler techniques using heuristics can be used to expand binary images. Many of the related issues, however, need further research. Some of these problems have been mentioned and are recapitulated in Section 6.4.

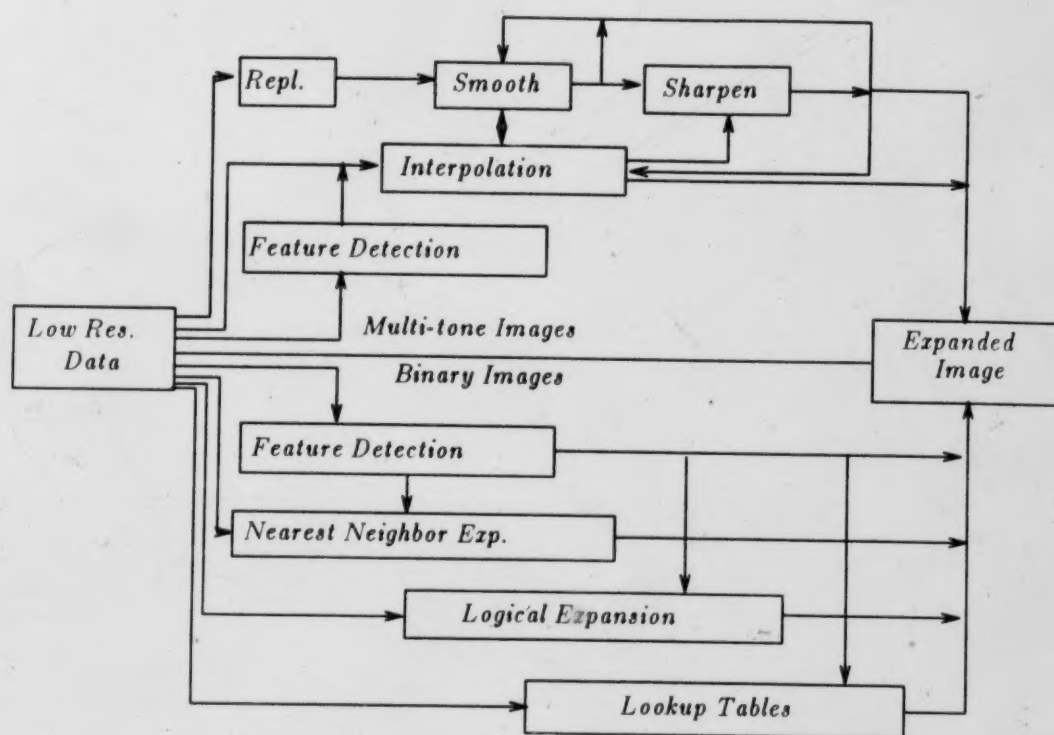


Figure 6.1 Block Diagram: Some techniques for expansion.

6.3. Application

From the results in Chapter 5, it is clear that none of the methods stands out in its performance. Unless the amount of blur can be quantified it is difficult to choose suitable ones for specific applications. On the other hand, if the main concern is to obtain a smooth expansion and the presence of some blur is not objectionable, bilinear interpolation is the most consistent in the sense that for almost any type of image content a reasonably smoothed output is produced. Based on this results, the use of the better methods for rendering low resolution data for browsing and windowing can provide faster and better high resolution display if the

availability of high resolution data takes longer than $O(k^2n^2)$ time. A combination of the methods might be experimented and used to produce anti-aliased images.

6.4. Further Research

It is clear that considerable work is still needed to provide more visually acceptable images. Apart from the pursuit of a proper model for human vision which will undoubtedly benefit all areas of image processing, the following problems require better understanding and research to enable better expansions.

1. A further study on the behavior of (2.90) to enable comparison between piecewise bicubic interpolation methods and the ideal interpolation filter. Other types of filters like the Gaussian can be studied as a comparison with the existing image interpolation methods.
3. Automatic thresholding in the averaging techniques. This will pose extra problems since it implies that the underlying nature of the image can be classified and features in an image can be distinguished as significant or otherwise.
4. Further experimentation to combine smoothing and sharpening techniques to suppress aliasing and simultaneously, highlight the certain set of edges and lines.
5. Partition the general expansion problem into more specific sub-classes like binary images so that more effective heuristics can be employed.
6. Investigate the feasibility of encoding low and high resolution data of an image such that areas with fewer intensity variations are stored in low resolution. Renditions of these images will then involve the conversion of the low resolution data incorporated with the other high resolution data.

7. With the availability of cheaper hardware and the parallel nature of the noise-cleaning and sharpening techniques, multi-processing adaptations of these techniques should be investigated to provide better time performance.

References

1. K.E. Atkinson, in *An Introduction to Numerical Analysis*, John Wiley & Sons, 1978.
2. R. Bracewell, in *The Fourier Transform and its Application*, McGraw Hill, 1978.
3. P.J. Burt and E.H. Adelson, "The Laplacian Pyramid as a Compact Code", *IEEE Trans. on Communications*, Vol. COM-31, 1983, pp. 532-540.
4. K.R. Castleman, in *Digital Image Processing*, Prentice-Hall, 1979.
5. W. Cheney and D. Kincaid, in *Numerical Mathematics and Computing*, Brooks/Cole, 1980.
6. F. Crow, "The Use of Gray Scale for Improved raster Display of Vectors and Characters.", *Comput. Graphics*, Vol. 12, No. 3, 1978, pp. 1-5.
7. J.P. Davenport, A Comparison of Noise-Cleaning Methods., Computing Science Technical Report 689, University of Maryland, College Park, Maryland, 1979.
8. L.S. Davis and A. Rosenfeld, "Noise Cleaning by Iterated Local Averaging.", *IEEE Trans. Syst., Man and Cybernetics*, Vol. SMC-8, 1978, pp. 705-710.
9. C. deBoor, in *A Practical Guide to Splines*, Springer-Verlag, 1978.
10. R.O. Duda and D.E. Hart, in *Pattern Recognition and Scene Analysis*, Wiley, 1973.
11. C.R. Dyer, A. Rosenfeld and H. Samet, "Boundary Codes from Quadrees", *Comm. ACM*, Vol. 23, No. 3, 1980, pp. 171-179.
12. D. Field, Two Algorithms for Drawing Anti-Aliasing Lines, *Graphics Interface '84*, pp. 87-98.

13. J.D. Foley and A. Van Dam, in *Fundamentals of Interactive Computer Graphics*, Addison-Wesley, 1983.
14. H. Freeman, Lines, Curves and The Characterization of Shape, *IFIP Congress 1980*, Tokyo, Japan and Melbourne, Australia, 1980, pp. 6-17.
15. Gonzalez and Wintz, in *Digital Image Processing*, Addison-Wesley, 1977.
16. G.H. Granlund, H. Knuttson and R. Wilson, Image Enhancement, in *Fundamentals in Computer Vision*, O.D. Faugeras (ed.), Cambridge Univ. Press, 1983.
17. R. Haralick, K. Shanmugam and J. Dinstein, "Textural features for Image Classification", *IEEE Trans. System, Man, Cybernetics*, Vol. SMC-3, No. 6, 1973, pp. 610-621.
18. E. Horowitz and S. Sahni, in *Fundamentals of Computer Algorithms*, Computer Science Press, 1978.
19. H.S. Hou and H.S. Andrews, "Cubic Splines for Interpolation and Filtering.", *IEEE Trans. Acous., Speech and Signal Processing*, Vol. ASSP-26, 1978, pp. 508-517.
20. T.S. Huang, G.J. Yang and G.Y. Tang, A Fast Two Dimensional Median Filtering Algorithm., *Proceeding Pattern Recognition and Image Processing*, Chicago, 1978, pp. 128-131.
21. R.A. Hummel, "Histogram Modification Techniques", *Comput. Graphics and Image Processing*, Vol. 4, 1975, pp. 209-224.
22. R.A. Hummel, "Image Enhancement by Histogram Transformation", *Comput. Graphics and Image Processing*, Vol. 6, 1977, pp. 184-195.
23. A. Lev, S.W. Zucker and A. Rosenfeld, "Iterative Enhancement of Noisy Images", *IEEE Trans. System, Man and Cybernetics*, Vol. SMC-7, No. 7, 1977,

pp. 435-442.

24. S. Levialdi, Edge Extraction Techniques, *Inrai-Crest Course on Computer Vision*, 1982, pp. 117-144.
25. Nagao and Matsuyama, "Edge Preserving Smoothing", *Comput. Graphics and Image Processing*, Vol. 9, 1977, pp. 394-407.
26. A. Naiman, Some New Ingredients for Cookbook Approach to Anti-Aliasing Text, *Graphics Interface '84*, pp. 99-108.
27. A. Netravali and F.W.Mounts, "Ordering Techniques for Facsimile Coding: A Review.", *Proc. IEEE*, Vol. 68, No. 7, 1980, pp. 796-807.
28. G. Ni and P. Tanner, "The Application of Anti-Aliasing Techniques for Displaying High Quality Chinese Characters.", *Computer Processing of Chinese and Oriental Languages*, Vol. 1, No. 1, July, 1983, pp. 71-77.
29. L.E. Ostrander, "The Fourier Transform of Spline Function Approximation to Continuous Data.", *IEEE Trans. Audio Electroacoustics.*, Vol. AU-19, 1971, pp. 103-104.
30. T. Pavlidis, in *Algorithms for Graphics and Image Processing*, Computer Science Press, 1982.
31. William K. Pratt, in *Digital Image Processing*, John-Wiley and Sons, 1978.
32. A. Rosenfeld, M. Thurston and Y.H. Lee, "Edge and Curve Detection: Further Experiments", *IEEE Trans. Comput.*, Vol. c-21, 1972, pp. 677-715.
33. A. Rosenfeld and A. Kak, in *Digital Picture Processing*, Vol. 1 and 2, Academic Press, 1982.
34. H. Samet, "Region Representation: Quadrees from Boundary Codes", *Comm. ACM*, Vol. 23, No. 3, 1980, pp. 163-170.

35. A. Scher, F.R. Velasco and A. Rosenfeld, "Some New Image Smoothing Techniques", *IEEE Trans. System, Man and Cybernetics*, Vol. SMC-10, 1980, pp. 153-158.
36. M.H. Schultz, in *Spline Analysis*, Prentice-Hall, 1973.
37. H. Spath, in *Spline Algorithms for Curves and Surfaces*, Unitas Mathematica Publ., 1974.
38. J.C. Stoffel, in *Graphical and Binary Image Processing and Applications*, Artech House, 1982.
39. G. Strang and G.J. Fix, in *An Analysis of the Finite Element Method*, Prentice-Hall, 1973.
40. F. Tomata and S. Tsuji, "Extraction of Multiple Regions by Smoothing in Selected Neighborhoods.", *IEEE Trans. System, Man and Cybernetics*, Vol. SMC-7, 1977, pp. 107-109.
41. R.A. Ulichney and D.E. Troxel, "Scaling Binary Images with Telescoping Template.", *IEEE Trans. Pattern and Machine Intelligence*, Vol. PAMI-4, 1982, pp. 331-335.
42. W.M. Newman and R.F. Sproull, in *Principles of Interactive Graphics*, McGraw Hill, 1979.